IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

WIREMED TECH LLC, Plaintiff, v.	C.A. NO. ______________ **JURY TRIAL DEMANDED**
ADOBE SYSTEMS INCORPORATED, Defendant.	**PATENT CASE**

ORIGINAL COMPLAINT FOR PATENT INFRINGEMENT AGAINST ADOBE SYSTEMS INCORPORATED

Plaintiff Wiremed Tech LLC files this Original Complaint for Patent Infringement against Adobe Systems Incorporated, and would respectfully show the Court as follows:

I. THE PARTIES

1. Plaintiff Wiremed Tech LLC ("Wiremed Tech" or "Plaintiff") is a Texas limited liability company with its principal place of business at 3000 Custer Road, Suite 270□7061, Plano, TX 75075.

2. On information and belief, Defendant Adobe Systems Incorporated ("Defendant") is a corporation organized and existing under the laws of Delaware, with a place of business at 345 Park Avenue, San Jose, CA 95110. Defendant has a registered agent at Corporation Service Company, 251 Little Falls Drive, Wilmington, DE 19808.

II. JURISDICTION AND VENUE

3. This action arises under the patent laws of the United States, Title 35 of the United States Code. This Court has subject matter jurisdiction of such action under 28 U.S.C. §§ 1331 and 1338(a).

4. On information and belief, Defendant is subject to this Court's specific and general personal jurisdiction, pursuant to due process and the Delaware Long-Arm Statute, due

at least to its business in this forum, including at least a portion of the infringements alleged herein. Furthermore, Defendant is subject to this Court's specific and general personal jurisdiction because Defendant is a Delaware corporation.

5. Without limitation, on information and belief, within this state, Defendant has used the patented inventions thereby committing, and continuing to commit, acts of patent infringement alleged herein. In addition, on information and belief, Defendant has derived revenues from its infringing acts occurring within Delaware. Further, on information and belief, Defendant is subject to the Court's general jurisdiction, including from regularly doing or soliciting business, engaging in other persistent courses of conduct, and deriving substantial revenue from goods and services provided to persons or entities in Delaware. Further, on information and belief, Defendant is subject to the Court's personal jurisdiction at least due to its sale of products and/or services within Delaware. Defendant has committed such purposeful acts and/or transactions in Delaware such that it reasonably should know and expect that it could be haled into this Court as a consequence of such activity.

6. Venue is proper in this district under 28 U.S.C. § 1400(b). On information and belief, Defendant is incorporated in Delaware. On information and belief, from and within this District Defendant has committed at least a portion of the infringements at issue in this case.

7. For these reasons, personal jurisdiction exists and venue is proper in this Court under 28 U.S.C. § 1400(b).

III. BACKGROUND

8. Plaintiff incorporates the above paragraphs herein by reference.

9. On December 18, 2001, United States Patent No. 6,331,864 ("the '864 Patent") was duly and legally issued by the United States Patent and Trademark Office. The application leading to the '864 patent was filed on Sept. 23, 1997. (Ex. A at cover). The '864 Patent is titled

"Real-Time Multimedia Visual Programming System." A true and correct copy of the '864 Patent is attached hereto as Exhibit A and incorporated herein by reference.

10. On September 13, 2005, United States Patent No. 6,944,825 ("the '825 Patent") was duly and legally issued by the United States Patent and Trademark Office. The application leading to the '825 patent was filed on October 9, 2001, and was a continuation of the application leading to the '864 patent. (Ex. B at cover). The '825 Patent is titled "Real-Time Multimedia Visual Programming System." A true and correct copy of the '825 Patent is attached hereto as Exhibit B and incorporated herein by reference.

11. Plaintiff is the assignee of all right, title and interest in the '864 Patent and the '825 Patent, including all rights to enforce and prosecute actions for infringement and to collect damages for all relevant times against infringers of the '864 Patent and the '825 Patent. Accordingly, Plaintiff possesses the exclusive right and standing to prosecute the present action for infringement of the '864 Patent and the '825 Patent by Defendant.

12. The inventions in the '864 Patent and '825 patent relate to the fields of computational control. (Ex. A at col. 1:5-12). Visual representations, such as the use of visual and auditory displays to represent information, has emerged as a desirable environment with which to effectively convey information because visual representation allows simultaneous presentation of multiple data streams. (*Id.* at col. 1:22-27).

13. Some general-purpose visualization systems include visual programming interfaces for users to create their own visualization environments from a library of program modules. (*Id.* at col. 1:28-31). The user must have a reasonably good understanding of software coding because a visual program module within a standard visual programming interface is a software-coded process consisting of a sequence of operations that are organized to produce a

specified result. (*Id.* at col. 1:31-36). The visual programming interfaces contain graphical tools for users to interactively assemble the modules into dataflow configurations of communication processes so that the user may visualize their data and create dataflow configurations. (*Id.* at col. 1:36-40). In other words, an output port of one module may be connected to the input port of another module so that when data flows between the modules, the receiving module's operations are performed. (*Id.* at col. 1:40-43). This requires that the user understand object-oriented programming structure and data representation techniques to connect and interpret connected modules. (*Id.* at col. 1:43-46). A typical dataflow configuration may therefore include a series of interconnected modules that are configured to read data, process the data into new representations, and then display the data on a computer screen. (*Id.* at col. 1:46-50).

14. Even though the general-purpose visual programming systems are flexible toward customizing dataflow configurations, they are not well suited for exploring real-time data streams' interactions, or creating multimedia experiences, because they lack essential interactive capabilities and require the user to have expert programming knowledge. (*Id.* at col. 1:51-57). For example, user interaction is typically limited to selecting how to display the data without any way to interactively manipulate the data at various stages of the data flow. (*Id.* at col. 1:57-60). As a result, the data can only be arranged in specific preconfigured visual arrangements. (*Id.* at col. 1:60-62).

15. In other words, known visual programming systems allow users to only visualize their data in pre-established ways. (*Id.* at col. 1:63-64). However, in order to fully experience their data, users must have real-time tools to probe the value of individual data elements as they impact a variety of parameters. (*Id.* at col. 1:67 – col. 2:3). They users must also be able to apply special procedure to a small region-of-interest, such as computing the mean for a subset of

data, and be able to visualize relationships between data elements in several images which may span several windows of a computer display. (*Id.* at col. 2:3-8).

16. In sum, there does not exist any visual programming tools that allow a person with little or no programming skills to create in real-time elaborate multimedia audio-visual episodes or human-machine interfaces with which anyone can interact using a multitude of input and output devices. (*Id.* at col. 2:9-14). The inventions of the '864 patent and '825 patent overcome these disadvantages by providing a visual programming interface that allows a user with no programming skills to create in real-time programs for exploring management control options or multimedia experiences. (*Id.* at col. 2:15-21).

17. The inventions allow a user to visually create multimedia programs in real-time. (*Id.* at col. 2:24-27). A user with no programming skills easily links desired transmit interface leads to desired receive interface leads. (*Id.* at col. 2:29-31). In real-time, the output devices perform the functions associated with receive interface leads and any changes to the receive interface leads that occur as a result of any links to transmit interface leads. (*Id.* at col. 2:31-34). The '864 patent provides several non-limiting examples of the claimed invention, which are described in relevant part below.

18. One example is for a visual programming system that allows a user with no programming skill to create a multi-media (*i.e.*, a combination of sound, graphics, animation, and video) computer program without writing linear text-based code which may then be re-executed as desired by the user. (*Id.* at col. 3:8-12). The system includes a processor, memory, a graphical user interface (GUI), and input and output devices coupled to the processor. (*Id.* at col. 3:13-16). The system allows interconnection between any input and output functions of input and output devices regardless of what those devices may be. (*Id.* at col. 3:16-19). Input devices

are dynamic data producing or generating devices, such as a mouse, joystick, touch pad, keyboard, motion sensor, or position sensor. (*Id.* at col. 3:19-25). Output devices react to produced data and include displays and robotics animatronics. (*Id.* at col. 3:25-29). The devices may be coupled to the processor locally or remotely though a local area or public data network (*e.g.*, the Internet). (*Id.* at col. 3:29-31).

19. An input device function identifies, in code (*e.g.*, a device driver), an operation or function performed by an input. (*Id.* at col. 3:35-37). For example, an input device can be a mouse with input functions of left button activate, move mouse vertically, move mouse horizontally, *etc*. (*Id.* at col. 3:38-41).

20. An output device function identified, in code (*e.g.*, a device driver), an operation or function performed by an output device. (*Id.* at col. 3:44-47). For example, an output device is a display device and examples of output functions include display a square, move the square vertically, move the square horizontally, *etc*. (*Id.* at col. 3:47-50). The display device receives or reacts to data that indicates what to display, where to display it, how to display it, *etc*. (*Id.* at col. 3:50-52).

21. The GUI allows the person with any level of programming experience to easily create a multimedia program by defining the relationship between any input device function and any output device function. (*Id.* at col. 3:53-57). An unskilled user can use the GUI to assemble and link an input device to an output device, in order to easily create and test a program in real-time. (*Id.* at col. 3:57-60).

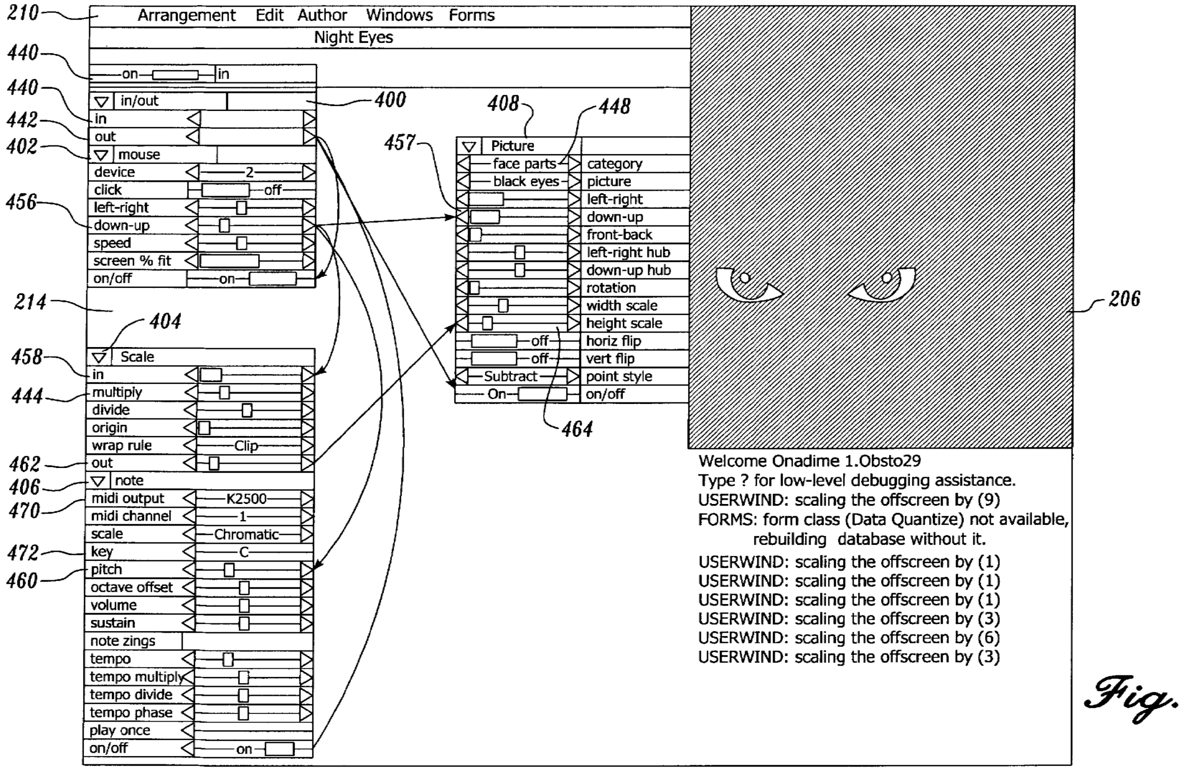
22. In the example, prior to unskilled user interaction with the claimed invention, the raw data translations used by input and output functions are determined by a more skilled programmer. (*Id.* at col. 3:62-65; Fig. 2). First, a skilled programmer selects a function and

determines if the selected function generates (*e.g.*, related to input) or reacts (*e.g.*, related to output) to raw data. (*Id.* at col. 4:3-10; Fig. 2). Then the skilled programmer determines the range of the raw data, which, for example, is determined by the raw data's upper and lower limit determined according to predefined requirements of the associated input or output device. (*Id.* at col. 4:10-14; Fig. 2). The skilled programmer then creates a translation calculation for translating the raw data into normalized data or normalized data into raw data. (*Id.* at col. 4:14-16; Fig. 2). Normalized data is a bound scalar number with a predesignated data range. (*Id.* at col. 4:16-18; Fig. 2). The '864 patent provides an exemplary equation for translating raw data into normalized data and vice versa. (*Id.* at col. 4:18-48). The scale of the normalized data is the same regardless of the raw data, which allows for simplification of a visual interface. (*Id.* at col. 4:49-52). Next, an interface lead is created for accessing a specific created translation calculation. (*Id.* at col. 4:56-59; Fig. 2). If the earlier selected function transmits data, it is a transmit interface lead; and if the earlier selected function instead reacts to data, it is a receive interface lead. (*Id.* at col. 4:59-63). The leads are then placed into sets of leads or forms that are stored in memory for easy retrieval and execution through the user interface. (*Id.* at col. 4:64-67; Fig. 2). The interaction of the leads within a form are defined using an algorithm or general function that depends upon the device the form represents. (*Id.* at col. 4:67 – col. 5:2; Fig. 2).

23. Upon the completion of storage of created interface leads and forms, a user with little or no programming skills can create a multimedia program by defining links between the created and stored input and receive interface leads using the GUI. (*Id.* at col. 5:10-14; Fig. 2). Because transmit interface leads represent functions that generate data and receive interface leads represent functions that react to data, only a transmit lead may be linked to a receive interface lead. (*Id.* at col. 5:14-17; Fig. 2). The interface leads may be linked, because of, for example,

the use of normalized data which allows the linking of leads without concern for raw data types or raw data numerical ranges. (*Id.* at col. 5:19-22).

24. An example of a GUI is shown in Figure 6 of the ‘864 patent:



```
Welcome Onadime 1.Obsto29
Type ? for low-level debugging assistance.
USERWIND: scaling the offscreen by (9)
FORMS: form class (Data Quantize) not available,
        rebuilding database without it.
USERWIND: scaling the offscreen by (1)
USERWIND: scaling the offscreen by (1)
USERWIND: scaling the offscreen by (1)
USERWIND: scaling the offscreen by (3)
USERWIND: scaling the offscreen by (6)
USERWIND: scaling the offscreen by (3)
```

Fig. 6

(Ex. A at Fig. 6). For example, if the output form represents a graphical image, such as a digital picture (*e.g.*, night eyes in Figure 6), the graphical image that is associated with the output form is presented in the display window in real-time. (*Id.* at col. 6:33-38). If the receive interface lead of the output has an arrow linking it to a transmit interface lead, the linked transmit interface lead translates raw data from an associated input function to normalized data; then the normalized data of the receive interface lead of the output form is translated into raw data, thereby changing the displayed graphical image. (*Id.* at col. 6:38-46).

25. For example, if the input device is a mouse and the down-up of the mouse is linked to the down-up movement of the night eyes within the display, when the mouse is moved

up, the raw data from the mouse is normalized and then translated into raw data for the display thereby resulting in the night eyes moving up in the display area in real-time. (*Id.* at col. 9:16-31). The user could the change the input device from a mouse to a motion or position sensor. Rather than moving a mouse, the motion or position sensor sense movement, the data from the sensor is translated into normalized data that is transmitted into raw data for the display thereby resulting in the night eyes moving in a manner corresponding to that sensed by the motion or position sensor. (*E.g.*, Ex. A at col. 9:63 – col. 10:2). The invention therefore reduces errors that result from making numerous changes before testing the modifications and increases the rate of experimentation and evolution. (*Id.* at col. 10:2-5).

26. The former owner of the patents-in-suit, Onadime, Inc., implemented the method through their product Onamation that, for example, allowed digital puppets, such as Bugs Bunny, to interact with a live improv voice to give live interviews, co-hosting live-action TV shows, and appearing "live" onstage. (*E.g.*, www.onadime.com/wiki/home).

IV. COUNT I
(PATENT INFRINGEMENT OF UNITED STATES PATENT NO. 6,331,864)

27. The foregoing paragraphs are incorporated by reference.

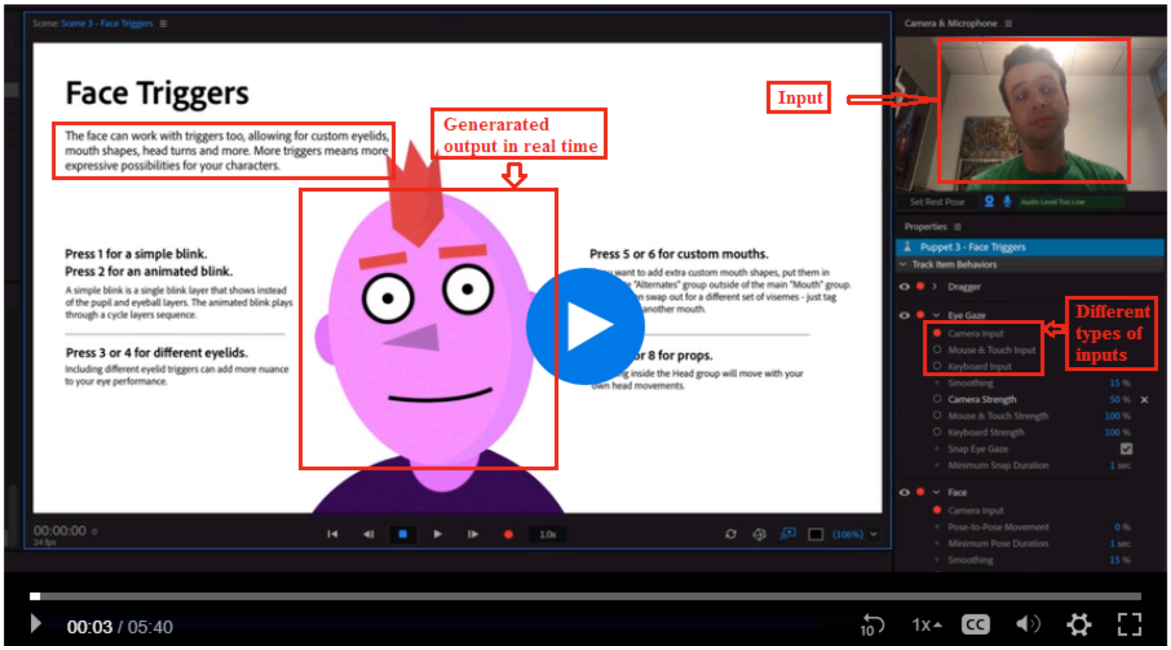
28. **Direct Infringement.** Upon information and belief, Defendant has been directly infringing claim of the '864 patent in Delaware, and elsewhere in the United States, by making, using, selling, and offering for sale software that satisfies each and every limitation of at least claims 3, 4, 5, 8, and 15 including without limitation the Adobe Character Animator ("Accused Instrumentality"). (*E.g.*, http://www.adobe.com/products/character-animator.html).

29. The Accused Instrumentality practices a method performed on a computer for interoperating data inputs (*e.g.*, animation selection data input) received from an input source (*e.g.*, a mouse, a keyboard, a webcam, or a microphone) and data outputs (*e.g.*, animation

generation for a character, image, or object). The Accused Instrumentality is an animator program, wherein the output of the animation program is dynamically generated in real time (*e.g.*, the visual display of animations created in the Accused Instrumentality are generated and executed in real-time based on real-time user input).

30. The Accused Instrumentality defines input functions based on the data inputs (*e.g.*, a user input function is a behavior to control animation functionalities in Adobe Character Animator through a mouse, a keyboard, or webcam). As shown below, different types of inputs (*e.g.*, camera input/ mouse & touch input/keyboard input) can be used to control the movement of the animated character on the visual display simultaneously. Thus, "left-right" transmit control representing the "left-right" movement of a mouse/the person's head creates an input function based on the data inputs that generates data that can be used for an output function.

Add triggers to a character's face.



Face Triggers

The face can work with triggers too, allowing for custom eyelids, mouth shapes, head turns and more. More triggers means more expressive possibilities for your characters.

Press 1 for a simple blink.
Press 2 for an animated blink.

A simple blink is a single blink layer that shows instead of the pupil and eyeball layers. The animated blink plays through a cycle layers sequence.

Press 3 or 4 for different eyelids.

Including different eyelid triggers can add more nuance to your eye performance.

Press 5 or 6 for custom mouths.

…u want to add extra custom mouth shapes, put them in …e "Alternates" group outside of the main "Mouth" group. …n swap out for a different set of visemes - just tag …another mouth.

…or 8 for props.

…g inside the Head group will move with your own head movements.

Puppet 3 - Face Triggers

Track Item Behaviors

Dragger

Eye Gaze

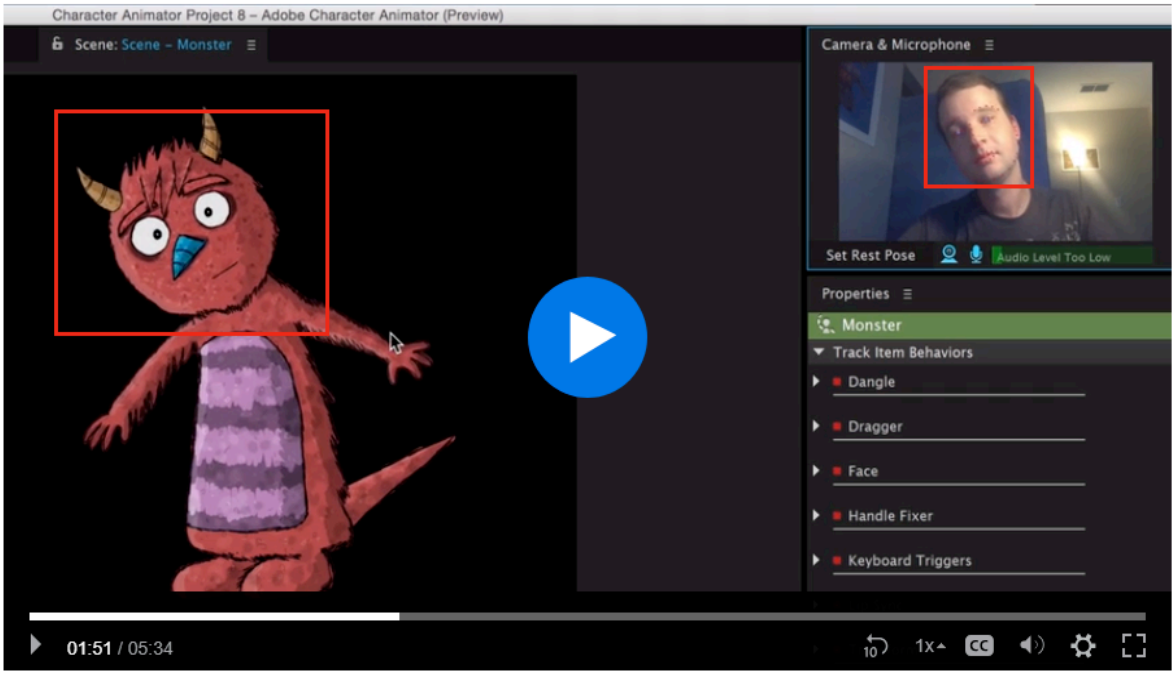
● Camera Input
○ Mouse & Touch Input
○ Keyboard Input
Smoothing 15 %
○ Camera Strength 50 %
○ Mouse & Touch Strength 100 %
○ Keyboard Strength 100 %
Snap Eye Gaze ☑
Minimum Snap Duration 1 sec

Face

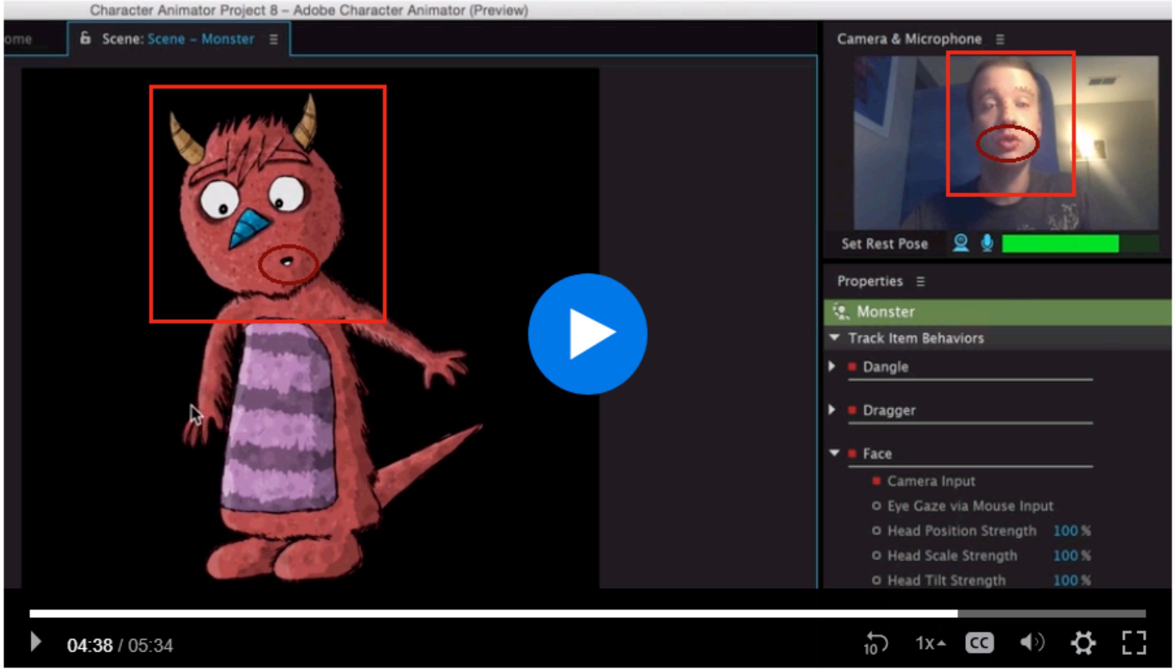
● Camera Input
Pose-to-Pose Movement 0 %
Minimum Pose Duration 1 sec
Smoothing 15 %

(https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.

adobe.com).

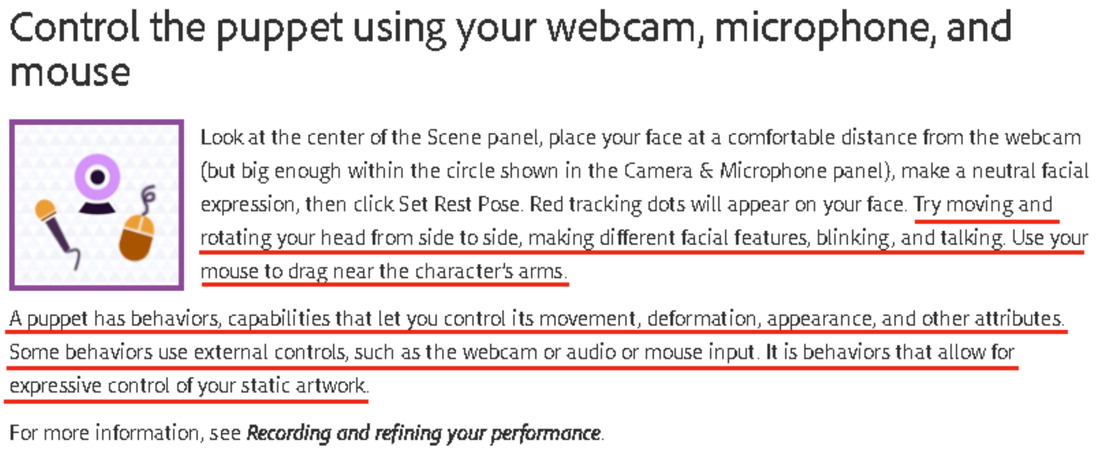


(https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com). As shown below, (i) “opening-closing” of a person’s mouth or (ii) sounds from talking into a microphone is input data that is converted by an input function into data that can be used for output.



Head Position Strength 100 %
Head Scale Strength 100 %
Head Tilt Strength 100 %

(https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com).



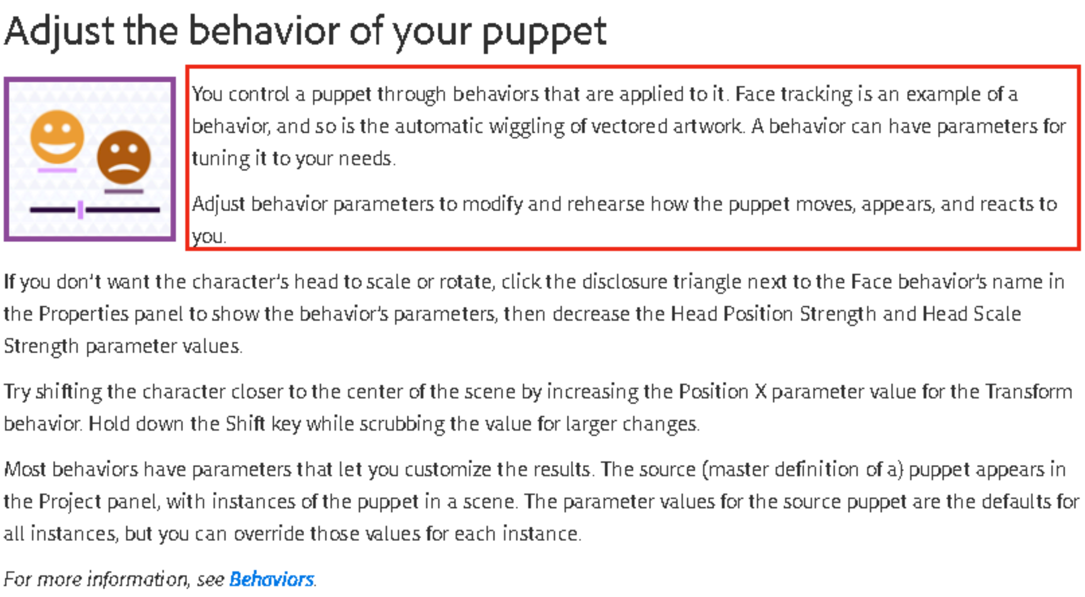
Control the puppet using your webcam, microphone, and mouse

Look at the center of the Scene panel, place your face at a comfortable distance from the webcam (but big enough within the circle shown in the Camera & Microphone panel), make a neutral facial expression, then click Set Rest Pose. Red tracking dots will appear on your face. Try moving and rotating your head from side to side, making different facial features, blinking, and talking. Use your mouse to drag near the character's arms.

A puppet has behaviors, capabilities that let you control its movement, deformation, appearance, and other attributes. Some behaviors use external controls, such as the webcam or audio or mouse input. It is behaviors that allow for expressive control of your static artwork.

For more information, see ***Recording and refining your performance***.

(https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol).



Adjust the behavior of your puppet

You control a puppet through behaviors that are applied to it. Face tracking is an example of a behavior, and so is the automatic wiggling of vectored artwork. A behavior can have parameters for tuning it to your needs.

Adjust behavior parameters to modify and rehearse how the puppet moves, appears, and reacts to you.

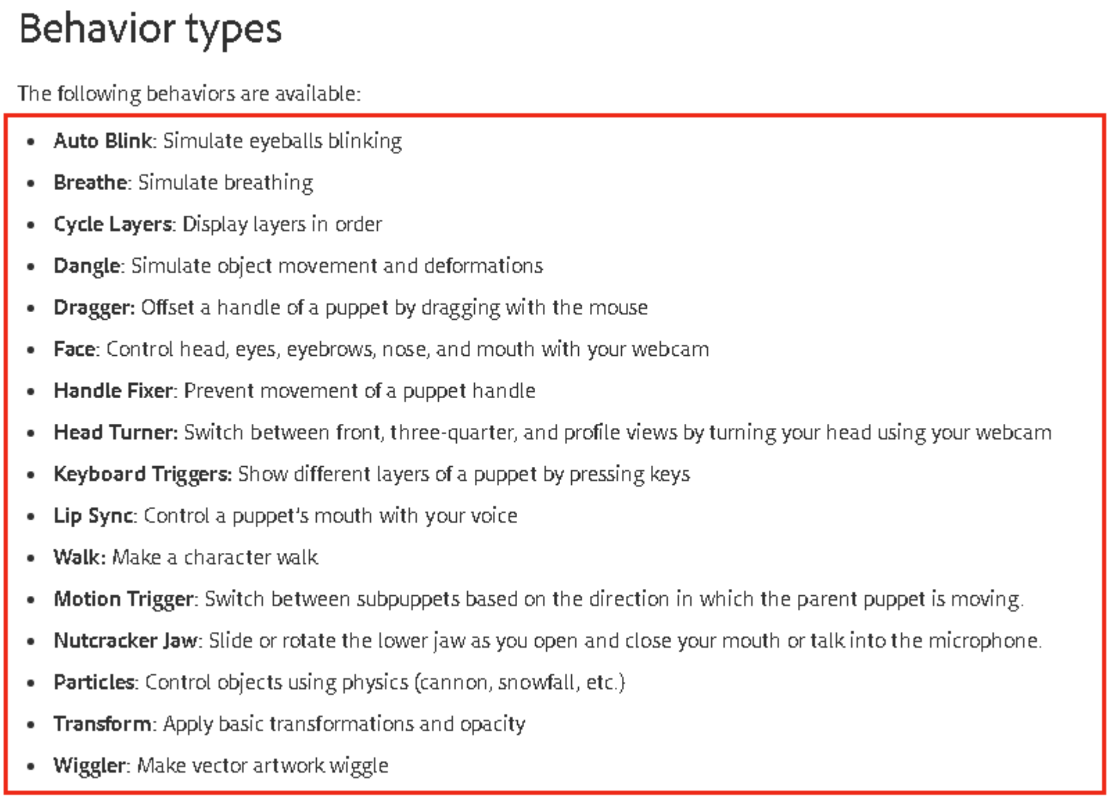
If you don't want the character's head to scale or rotate, click the disclosure triangle next to the Face behavior's name in the Properties panel to show the behavior's parameters, then decrease the Head Position Strength and Head Scale Strength parameter values.

Try shifting the character closer to the center of the scene by increasing the Position X parameter value for the Transform behavior. Hold down the Shift key while scrubbing the value for larger changes.

Most behaviors have parameters that let you customize the results. The source (master definition of a) puppet appears in the Project panel, with instances of the puppet in a scene. The parameter values for the source puppet are the defaults for all instances, but you can override those values for each instance.

For more information, see ***Behaviors****.*

(https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol).



Behavior types

The following behaviors are available:

- **Auto Blink**: Simulate eyeballs blinking
- **Breathe**: Simulate breathing
- **Cycle Layers**: Display layers in order
- **Dangle**: Simulate object movement and deformations
- **Dragger:** Offset a handle of a puppet by dragging with the mouse
- **Face**: Control head, eyes, eyebrows, nose, and mouth with your webcam
- **Handle Fixer**: Prevent movement of a puppet handle
- **Head Turner:** Switch between front, three-quarter, and profile views by turning your head using your webcam
- **Keyboard Triggers:** Show different layers of a puppet by pressing keys
- **Lip Sync**: Control a puppet's mouth with your voice
- **Walk:** Make a character walk
- **Motion Trigger**: Switch between subpuppets based on the direction in which the parent puppet is moving.
- **Nutcracker Jaw**: Slide or rotate the lower jaw as you open and close your mouth or talk into the microphone.
- **Particles**: Control objects using physics (cannon, snowfall, etc.)
- **Transform**: Apply basic transformations and opacity
- **Wiggler**: Make vector artwork wiggle

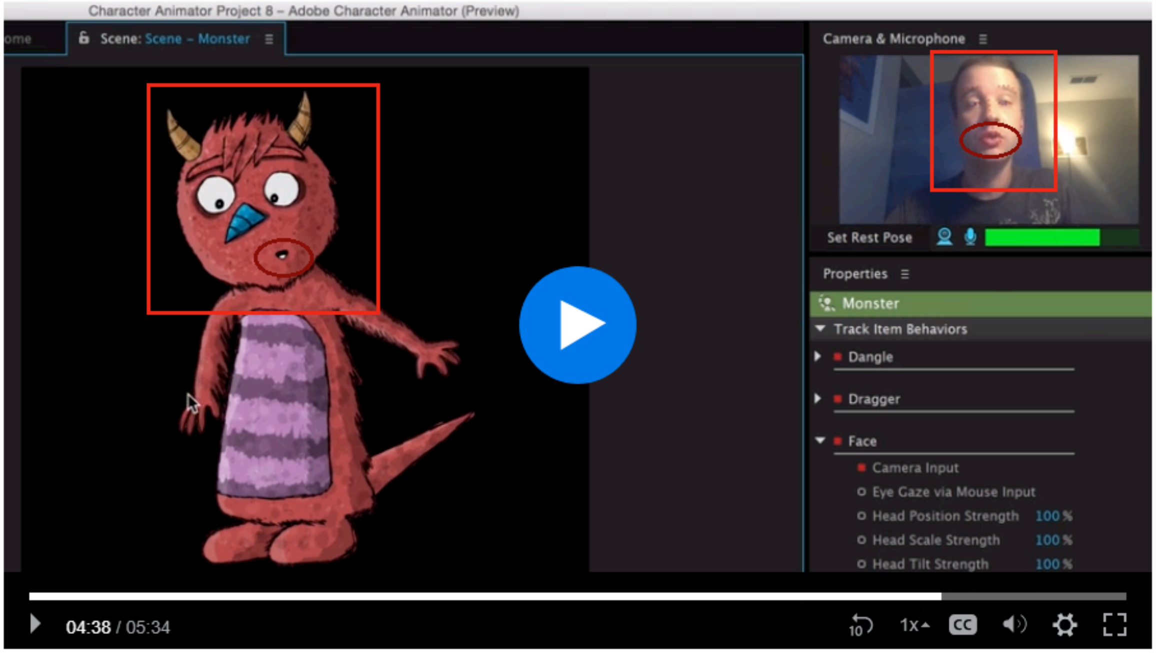
(https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol).

31. The Accused Instrumentality defines output functions (*e.g.*, animation generation function for an image, an object, a character, etc. implementing a certain animation effect related to the character's movement, appearance) based on the data outputs (*e.g.*, control data received by a desktop display corresponding to animation output). (*Supra* ¶30). The output functions include how to move a displayed faced based on data received and define how the animation moves based on the data received. (*Id.*).

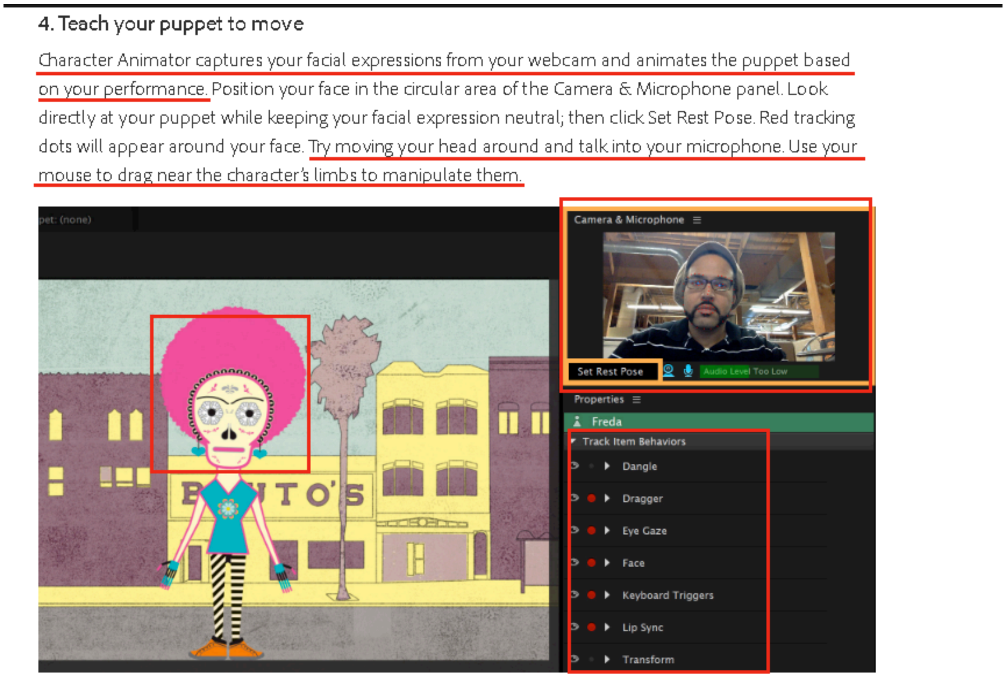
32. The Accused Instrumentality links at least one input function to at least one output function (*e.g.*, a character's behaviors let the user control its movement, deformation, appearance, and other attributes) to indicate that the at least one output function will change as the linked at least one input function changes, and without determining if the type of data input (*e.g.*, a user input through a mouse, a keyboard, webcam, microphone) upon which the at least one input function (*e.g.*, a user input function (*i.e.*, a behavior to control animation functionalities through a mouse, a keyboard, webcam)) is based is compatible with the type of data output (*e.g.*, control data received by a desktop display corresponding to animation output) upon which the at least one output (*e.g.*, character's movement, deformation, appearance) is based. (*e.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com;

https://helpx.adobe.com/adobe-character-animator/how-to/animate-puppet-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com). In other words, two different types of inputs that have different input data can result in the same data output by the output function. Different types of inputs, *e.g.*, camera input/mouse & touch input/keyboard input can be used to control the movement of the animated character on the visual display simultaneously. (*Id.*). Thus, "left-right" transmit control representing the "left-right" movement of a mouse/the person's head linked to a "left-right" receive control representing the "left-right" movement of an image on the visual display simultaneously is indicative of the fact that the output is displayed by defining a relationship between said at least one transmit control and said at least one receive control without determining if the type of data input to said animation program is compatible with the type of output from said animation program. (*Id.*).

33. For example, (i) "opening-closing" of a person's mouth is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of the animated mouth on display or (ii) sounds from talking into a microphone is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display; because either input can be used, it is using normalized data, and there is live output. Thus, either opening or closing your mouth or talking into a microphone moves the jaw; again, each of those inputs are converted into normalized data so that it can be used to animate the mouth.



(https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com; *also* https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol).



4. Teach your puppet to move

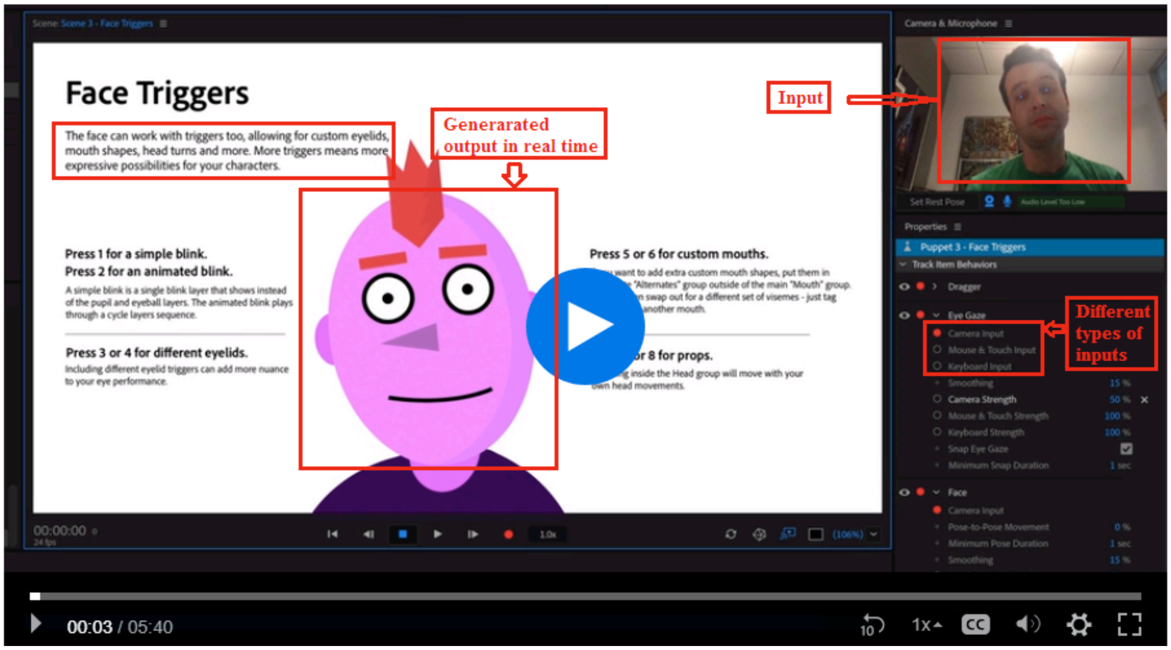
Character Animator captures your facial expressions from your webcam and animates the puppet based on your performance. Position your face in the circular area of the Camera & Microphone panel. Look directly at your puppet while keeping your facial expression neutral; then click Set Rest Pose. Red tracking dots will appear around your face. Try moving your head around and talk into your microphone. Use your mouse to drag near the character's limbs to manipulate them.

(https://helpx.adobe.com/adobe-character-animator/how-to/adobe-character-animator.html).

34. The Accused Instrumentality practices dynamically controlling the at least one output function (*e.g.*, animation generation function for a character implementing a certain animation effect related to the character's movement, appearance) by changing the at least one input function (*e.g.*, a user input function such as a behavior to control animation functionalities through mouse, keyboard, webcam, microphone), wherein changing the linked at least one input function (*e.g.*, facial expression of the user in case of face tracking, voice of the user in case of lip-sync, mouse movement in case of limb motion) and controlling the at least one output function (*e.g.*, animation generation function for movement of the face, mouth, limbs of the character) occur simultaneously.

35. Different types of inputs (*e.g.*, camera input/mouse & touch input/keyboard input) can be used to control the movement of the animated character on the visual display simultaneously. Thus, "left-right" transmit control representing the "left-right" movement of a mouse/the person's head linked to a "left-right" receive control representing the "left-right" movement of an image on the visual display simultaneously is indicative of the fact that the output is displayed defining a relationship between said at least one transmit control and said at least one receive control without determining if the type of data input to said animation program is compatible with the type of output from said animation program. Furthermore, as shown below, the output generated along with inputs in real time indicate live output.

Add triggers to a character's face.



Face Triggers

The face can work with triggers too, allowing for custom eyelids, mouth shapes, head turns and more. More triggers means more expressive possibilities for your characters.

Press 1 for a simple blink.
Press 2 for an animated blink.

A simple blink is a single blink layer that shows instead of the pupil and eyeball layers. The animated blink plays through a cycle layers sequence.

Press 3 or 4 for different eyelids.

Including different eyelid triggers can add more nuance to your eye performance.

Press 5 or 6 for custom mouths.

want to add extra custom mouth shapes, put them in "Alternates" group outside of the main "Mouth" group. swap out for a different set of visemes - just tag another mouth.

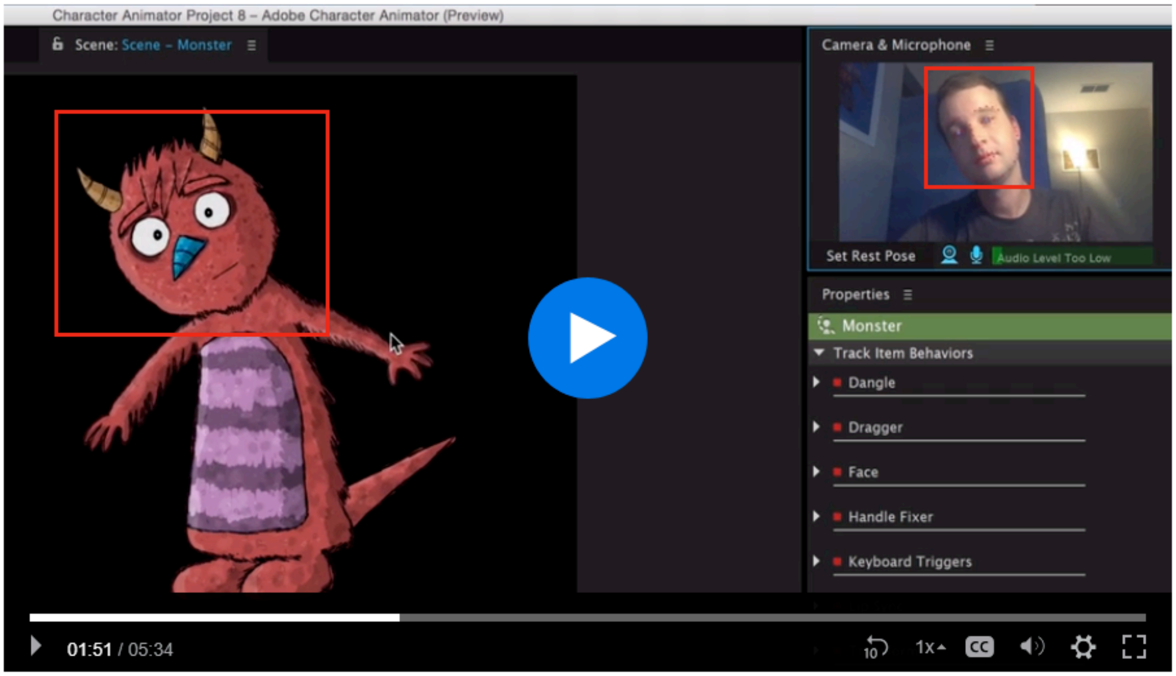
or 8 for props.

inside the Head group will move with your own head movements.

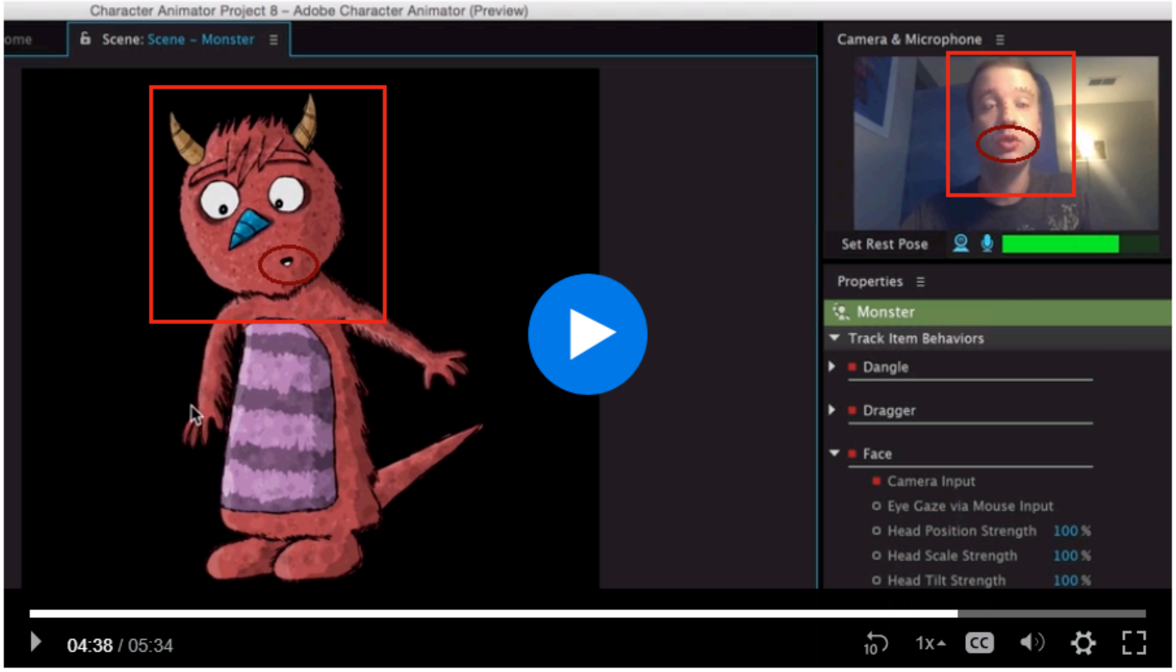
Eye Gaze: ● Camera Input, ○ Mouse & Touch Input, ○ Keyboard Input; Smoothing 15 %; ○ Camera Strength 50 %; ○ Mouse & Touch Strength 100 %; ○ Keyboard Strength 100 %; Snap Eye Gaze ☑; Minimum Snap Duration 1 sec

Face: ● Camera Input; Pose-to-Pose Movement 0 %; Minimum Pose Duration 1 sec; Smoothing 15 %

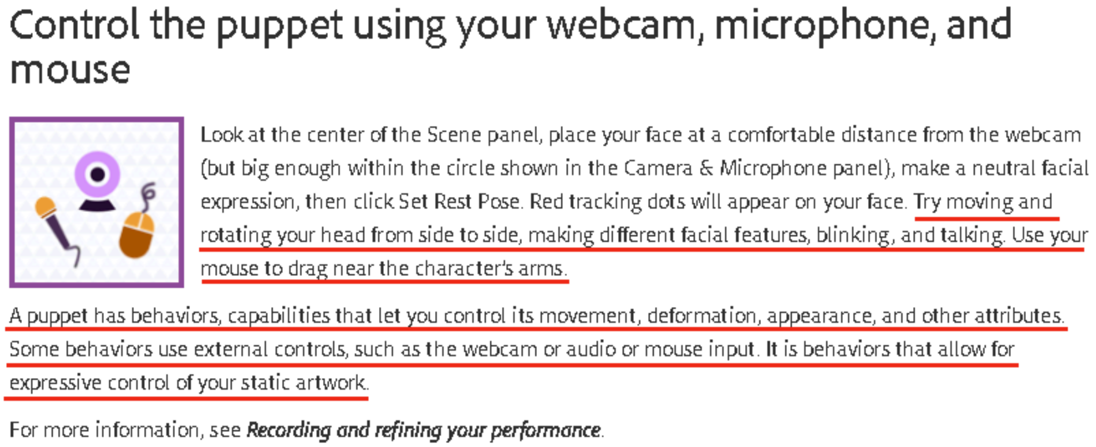
(https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com).



(https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com).



(https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com).



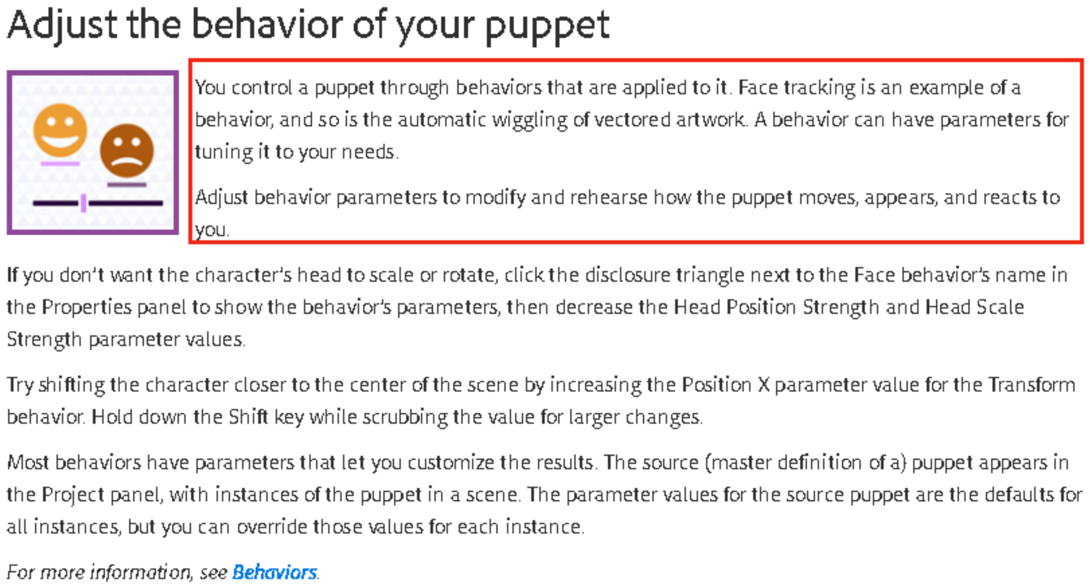
Control the puppet using your webcam, microphone, and mouse

Look at the center of the Scene panel, place your face at a comfortable distance from the webcam (but big enough within the circle shown in the Camera & Microphone panel), make a neutral facial expression, then click Set Rest Pose. Red tracking dots will appear on your face. Try moving and rotating your head from side to side, making different facial features, blinking, and talking. Use your mouse to drag near the character's arms.

A puppet has behaviors, capabilities that let you control its movement, deformation, appearance, and other attributes. Some behaviors use external controls, such as the webcam or audio or mouse input. It is behaviors that allow for expressive control of your static artwork.

For more information, see ***Recording and refining your performance***.

https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol



Adjust the behavior of your puppet

You control a puppet through behaviors that are applied to it. Face tracking is an example of a behavior, and so is the automatic wiggling of vectored artwork. A behavior can have parameters for tuning it to your needs.

Adjust behavior parameters to modify and rehearse how the puppet moves, appears, and reacts to you.

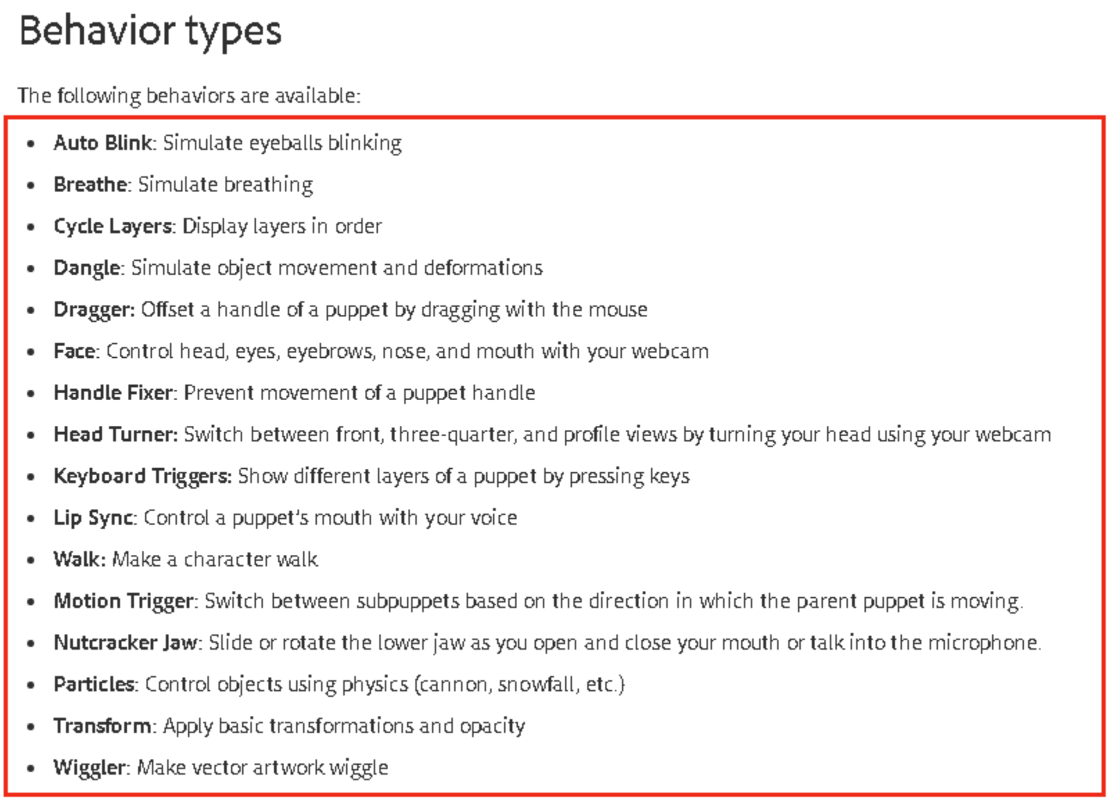
If you don't want the character's head to scale or rotate, click the disclosure triangle next to the Face behavior's name in the Properties panel to show the behavior's parameters, then decrease the Head Position Strength and Head Scale Strength parameter values.

Try shifting the character closer to the center of the scene by increasing the Position X parameter value for the Transform behavior. Hold down the Shift key while scrubbing the value for larger changes.

Most behaviors have parameters that let you customize the results. The source (master definition of a) puppet appears in the Project panel, with instances of the puppet in a scene. The parameter values for the source puppet are the defaults for all instances, but you can override those values for each instance.

*For more information, see **Behaviors**.*

https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol

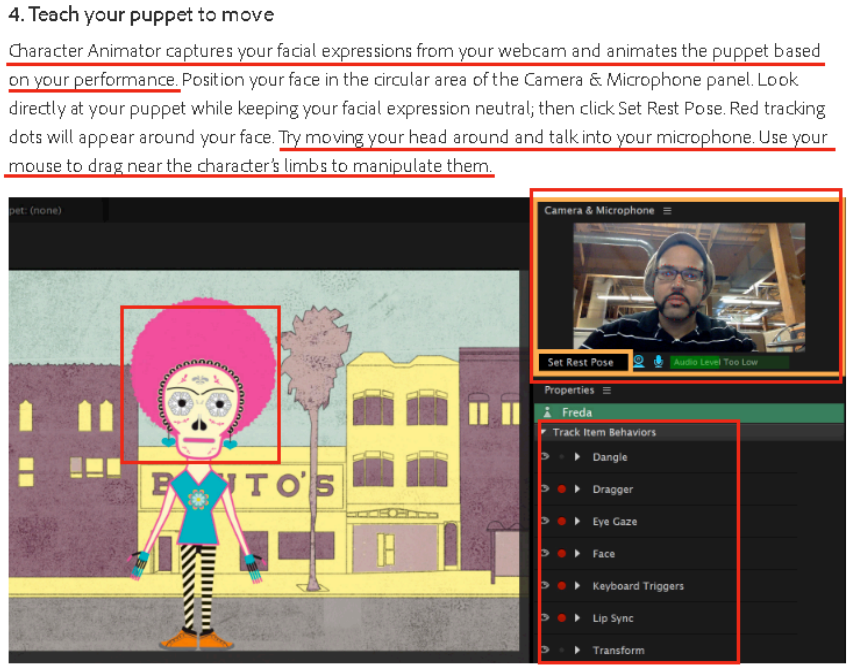


Behavior types

The following behaviors are available:

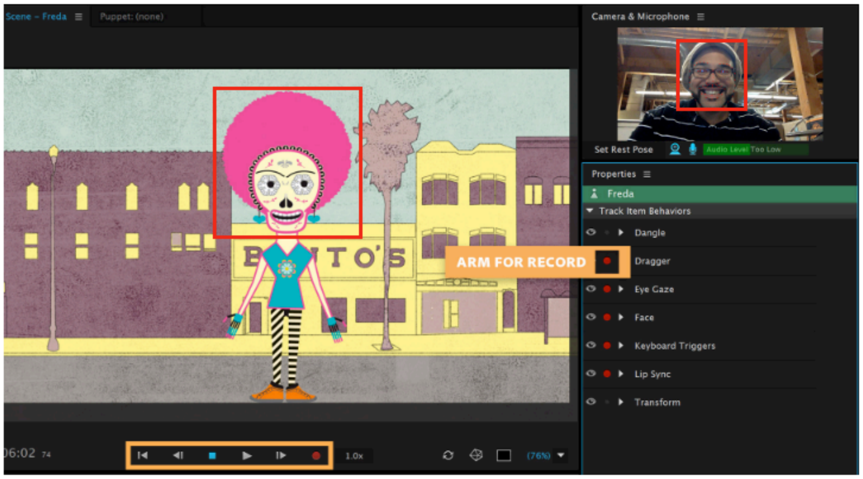
- **Auto Blink**: Simulate eyeballs blinking
- **Breathe**: Simulate breathing
- **Cycle Layers**: Display layers in order
- **Dangle**: Simulate object movement and deformations
- **Dragger:** Offset a handle of a puppet by dragging with the mouse
- **Face**: Control head, eyes, eyebrows, nose, and mouth with your webcam
- **Handle Fixer**: Prevent movement of a puppet handle
- **Head Turner:** Switch between front, three-quarter, and profile views by turning your head using your webcam
- **Keyboard Triggers:** Show different layers of a puppet by pressing keys
- **Lip Sync**: Control a puppet's mouth with your voice
- **Walk:** Make a character walk
- **Motion Trigger**: Switch between subpuppets based on the direction in which the parent puppet is moving.
- **Nutcracker Jaw**: Slide or rotate the lower jaw as you open and close your mouth or talk into the microphone.
- **Particles**: Control objects using physics (cannon, snowfall, etc.)
- **Transform**: Apply basic transformations and opacity
- **Wiggler**: Make vector artwork wiggle

https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol

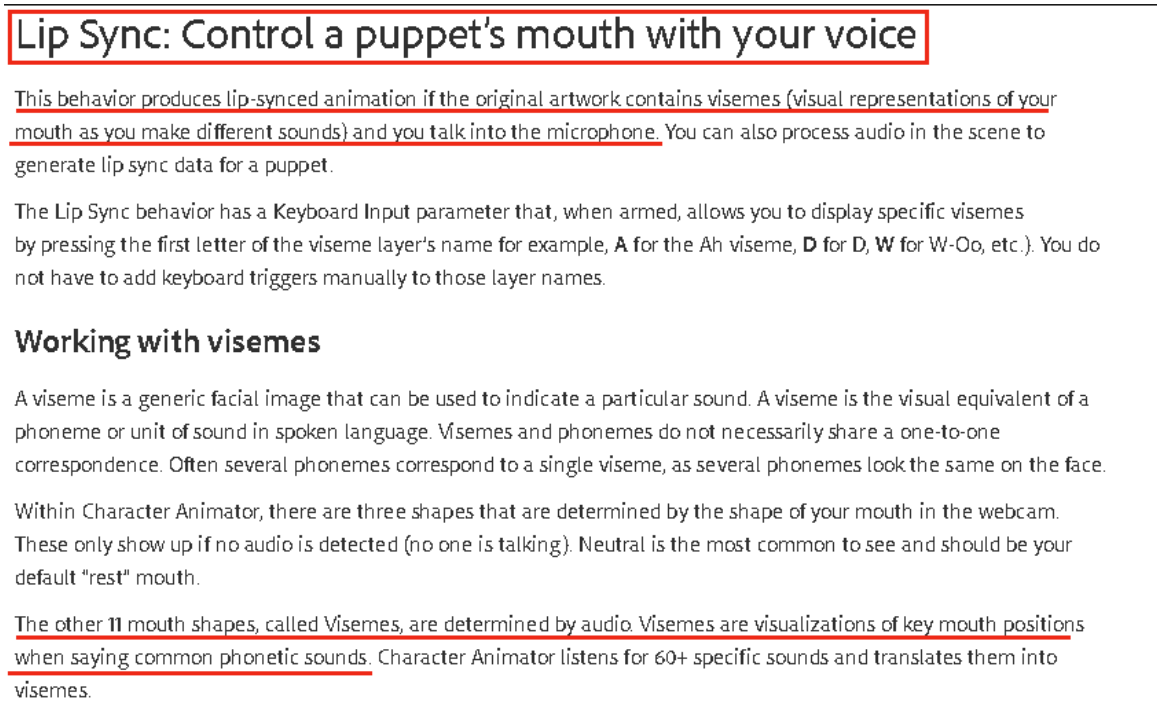


4. Teach your puppet to move

Character Animator captures your facial expressions from your webcam and animates the puppet based on your performance. Position your face in the circular area of the Camera & Microphone panel. Look directly at your puppet while keeping your facial expression neutral; then click Set Rest Pose. Red tracking dots will appear around your face. Try moving your head around and talk into your microphone. Use your mouse to drag near the character's limbs to manipulate them.



(https://helpx.adobe.com/adobe-character-animator/how-to/adobe-character-animator.html).



Lip Sync: Control a puppet's mouth with your voice

This behavior produces lip-synced animation if the original artwork contains visemes (visual representations of your mouth as you make different sounds) and you talk into the microphone. You can also process audio in the scene to generate lip sync data for a puppet.

The Lip Sync behavior has a Keyboard Input parameter that, when armed, allows you to display specific visemes by pressing the first letter of the viseme layer's name for example, **A** for the Ah viseme, **D** for D, **W** for W-Oo, etc.). You do not have to add keyboard triggers manually to those layer names.

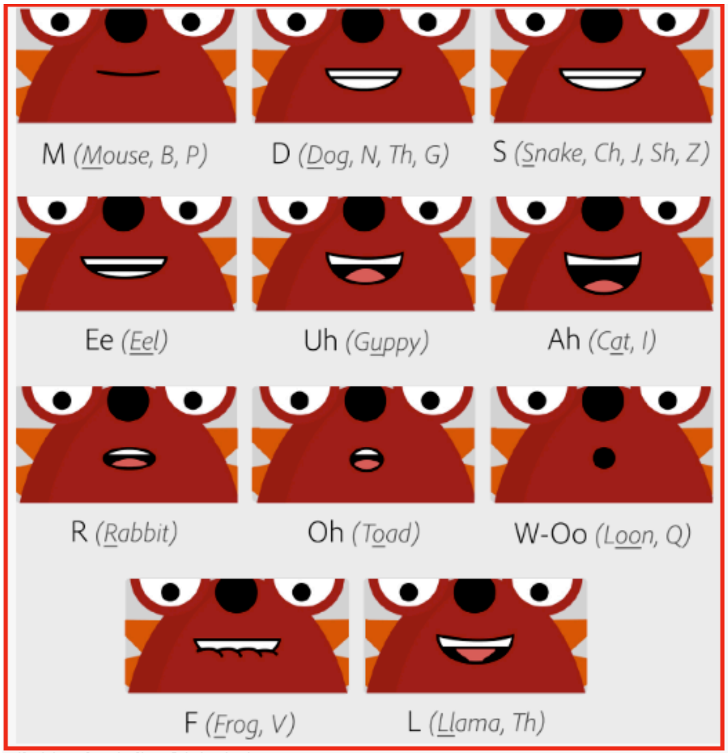
Working with visemes

A viseme is a generic facial image that can be used to indicate a particular sound. A viseme is the visual equivalent of a phoneme or unit of sound in spoken language. Visemes and phonemes do not necessarily share a one-to-one correspondence. Often several phonemes correspond to a single viseme, as several phonemes look the same on the face.

Within Character Animator, there are three shapes that are determined by the shape of your mouth in the webcam. These only show up if no audio is detected (no one is talking). Neutral is the most common to see and should be your default "rest" mouth.

The other 11 mouth shapes, called Visemes, are determined by audio. Visemes are visualizations of key mouth positions when saying common phonetic sounds. Character Animator listens for 60+ specific sounds and translates them into visemes.

(https://helpx.adobe.com/adobe-character-animator/using/behavior.html#main-pars_header_40).



Mouthshapes determined by audio in the microphone

(https://helpx.adobe.com/adobe-character-animator/using/behavior.html#main-pars_header_40).

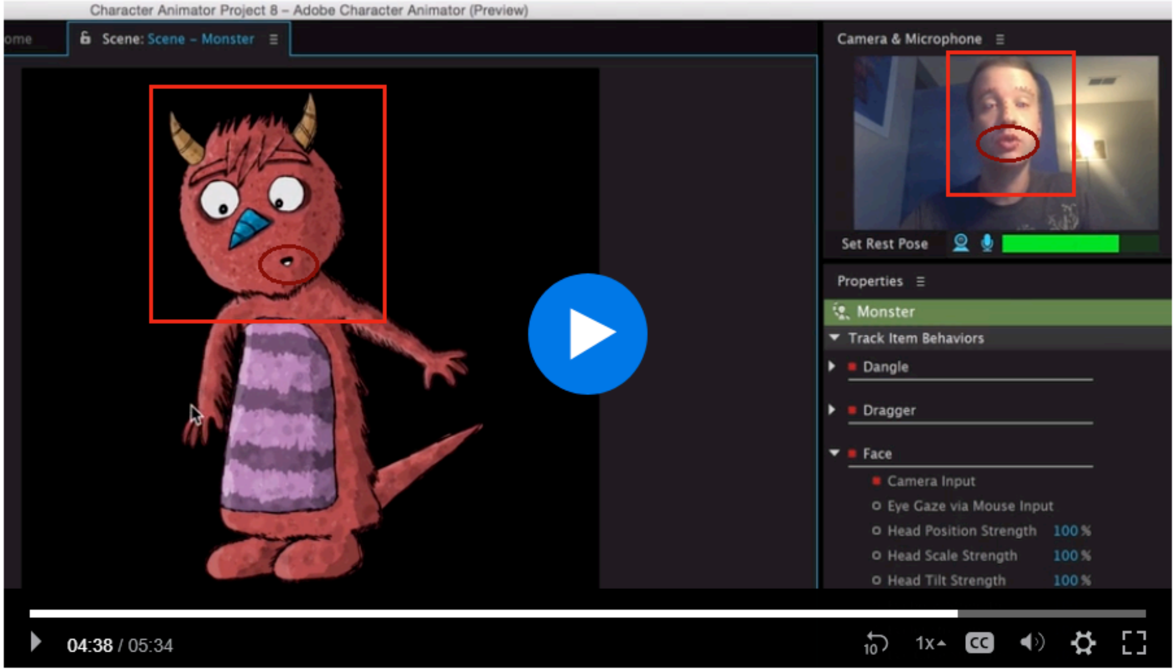
36. With respect to claim 4, the Accused Instrumentality practices defining the input functions and the output functions by: defining a normalized data range for data values assigned to the data inputs and the data outputs; defining an input function raw data range for data values generated by an input function of a data input; converting the data value generated by the input function into a data value within the normalized data range based on the input function raw data range; defining an output function raw data range for data values received by an output function of a data output; and converting the data value assigned to the data output into a data value within the output function raw data range based on the normalized data range. As shown above, different types of inputs (*e.g.*, camera input/mouse & touch input/keyboard input) can be used to control the movement of the animated character on the visual display simultaneously. Thus, "left-right" transmit control representing the "left-right" movement of a mouse/the person's head linked to a "left-right" receive control representing the "left-right" movement of an image on the

visual display simultaneously is indicative of the fact that the output is displayed defining a relationship between said at least one transmit control and said at least one receive control without determining if the type of data input to said animation program is compatible with the type of output from said animation program. As another example shown above, (i) "opening-closing" of a person's mouth is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display or (ii) sounds from talking into a microphone is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display; because either input can be used, it is using normalizing the data value of the input function and converting the normalized data value into a data value within the output function raw data range, and there is live output. Thus, either opening or closing your mouth or talking into a microphone moves the jaw; again, each of those inputs are converted into normalized data so that it can be used to animate the mouth and stay within the mouth range movements of the animated character. (*e.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com; http://www.adobe.com/products/character-animator.html; https://helpx.adobe.com/adobe-character-animator/how-to/adobe-character-animator.html; https://helpx.adobe.com/adobe-character-animator/using/behavior.html#main-pars_header_40).

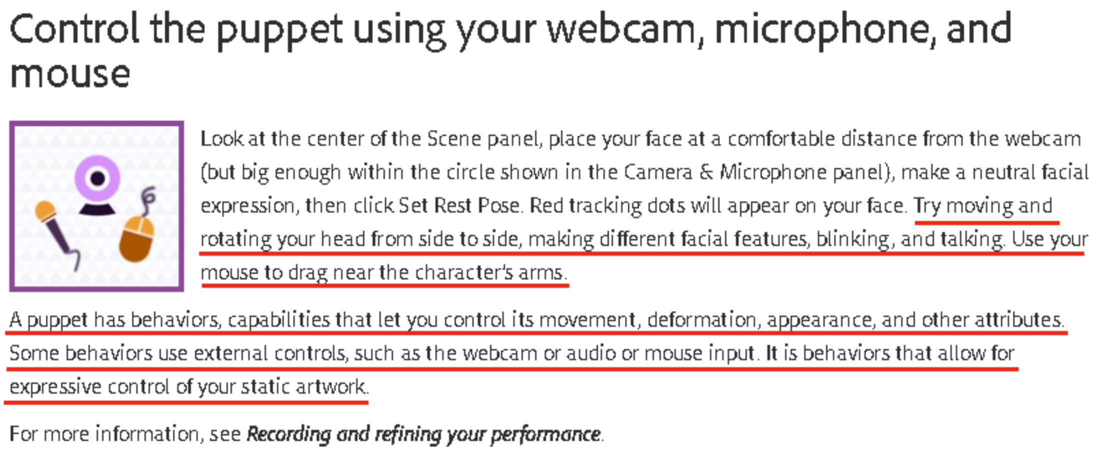
37. With respect to claim 5, the Accused Instrumentality practices defining a relationship between the defined normalized data values of the data input and the data output; and setting the normalized data value of the data output based on the normalized data value of the data input and the defined relationship between the defined normalized data values of the

data input and the data output. As shown above, different types of inputs, e.g., camera input/mouse & touch input/ keyboard input can be used to control the movement of the animated character on the visual display simultaneously. Thus, "left-right" transmit control representing the "left-right" movement of a mouse/the person's head linked to a "left-right" receive control representing the "left-right" movement of an image on the visual display simultaneously is indicative of the fact that the output is displayed defining a relationship between said at least one transmit control and said at least one receive control without determining if the type of data input to said animation program is compatible with the type of output from said animation program. Again, as shown above, (i) "opening-closing" of a person's mouth is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display or (ii) sounds from talking into a microphone is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display; because either input can be used, it is using normalized data, and there is live output. Thus, either opening or closing your mouth or talking into a microphone moves the jaw; again, each of those inputs are converted into normalized data so that it can be used to animate the mouth.

38. With respect to claim 8, the Accused Instrumentality practices that the data input comprises a visual data input (e.g., camera & microphone input). For example:



(https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com).



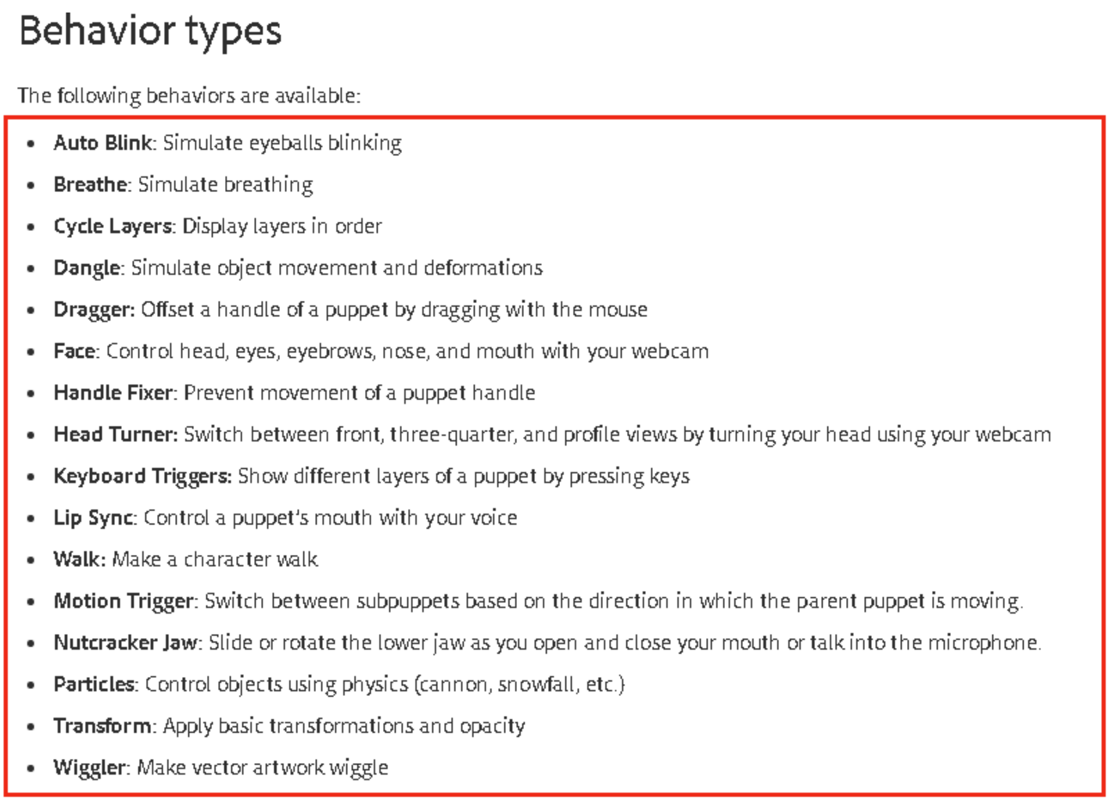
Control the puppet using your webcam, microphone, and mouse

Look at the center of the Scene panel, place your face at a comfortable distance from the webcam (but big enough within the circle shown in the Camera & Microphone panel), make a neutral facial expression, then click Set Rest Pose. Red tracking dots will appear on your face. Try moving and rotating your head from side to side, making different facial features, blinking, and talking. Use your mouse to drag near the character's arms.

A puppet has behaviors, capabilities that let you control its movement, deformation, appearance, and other attributes. Some behaviors use external controls, such as the webcam or audio or mouse input. It is behaviors that allow for expressive control of your static artwork.

For more information, see ***Recording and refining your performance***.

(https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol).

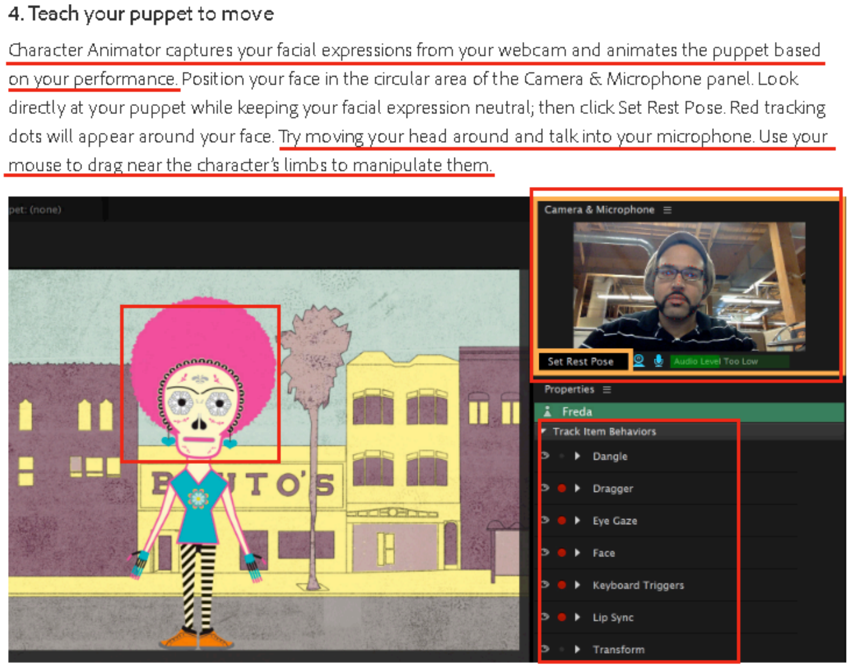


Behavior types

The following behaviors are available:

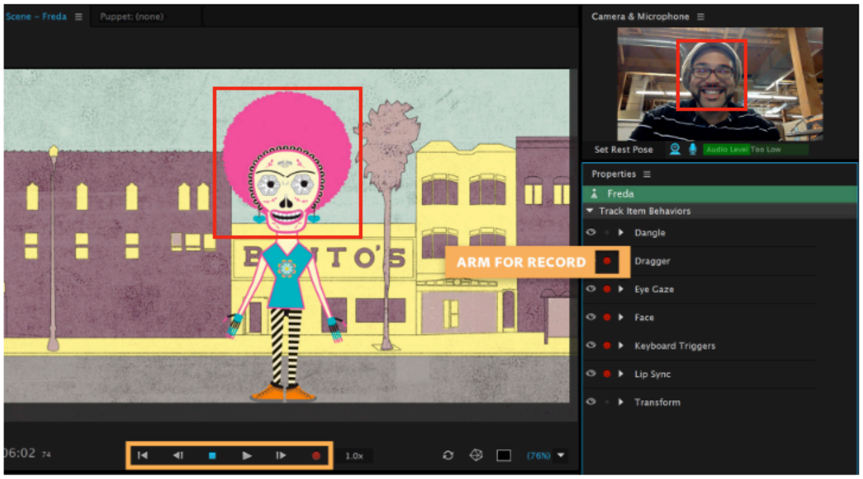
- **Auto Blink**: Simulate eyeballs blinking
- **Breathe**: Simulate breathing
- **Cycle Layers**: Display layers in order
- **Dangle**: Simulate object movement and deformations
- **Dragger:** Offset a handle of a puppet by dragging with the mouse
- **Face**: Control head, eyes, eyebrows, nose, and mouth with your webcam
- **Handle Fixer**: Prevent movement of a puppet handle
- **Head Turner:** Switch between front, three-quarter, and profile views by turning your head using your webcam
- **Keyboard Triggers:** Show different layers of a puppet by pressing keys
- **Lip Sync**: Control a puppet's mouth with your voice
- **Walk:** Make a character walk
- **Motion Trigger**: Switch between subpuppets based on the direction in which the parent puppet is moving.
- **Nutcracker Jaw**: Slide or rotate the lower jaw as you open and close your mouth or talk into the microphone.
- **Particles**: Control objects using physics (cannon, snowfall, etc.)
- **Transform**: Apply basic transformations and opacity
- **Wiggler**: Make vector artwork wiggle

(https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol).



4. Teach your puppet to move

Character Animator captures your facial expressions from your webcam and animates the puppet based on your performance. Position your face in the circular area of the Camera & Microphone panel. Look directly at your puppet while keeping your facial expression neutral; then click Set Rest Pose. Red tracking dots will appear around your face. Try moving your head around and talk into your microphone. Use your mouse to drag near the character's limbs to manipulate them.



(https://helpx.adobe.com/adobe-character-animator/how-to/adobe-character-animator.html).

39. With respect to claim 15, the Accused Instrumentality is a computer program stored on a computer-readable medium for programming the interoperability of input and output devices. The Accused Instrumentality is an animator program embodied on a computer-readable medium (*e.g.*, the Accused Instrumentality is software that is stored and operates on personal computers), wherein the output of said animation program is dynamically generated in real time (*e.g.*, the visual display of animations created in the Accused Instrumentality are generated and executed in real-time based on real-time user input). The Accused Instrumentality discloses a first defining component for defining input functions of the input devices (*e.g.*, defining camera & microphone/ keyboard/mouse & touch to act as input giving data for motion control of the animated character) as transmit controls; a second defining component for defining output functions of the output devices as receive controls (*e.g.*, defining the display of the desktop to display the corresponding motion control change of the animated character as inputted by the input data). (*e.g.*, http://www.adobe.com/products/character-animator.html; https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-

animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com).

40. The Accused Instrumentality discloses a first linking component for linking at least one transmit control to at least one receive control without determining if the data processed by the input devices is compatible with the data processed by the output devices, such that the at least one output function defined by the at least one receive control is dynamically controlled in real-time by manipulating the input function defined by the at least one transmit control linked to the at least one receive control. Different types of inputs (*e.g.*, camera input/mouse & touch input/ keyboard input) can be used to control the movement of the animated character on the visual display simultaneously. Thus, "left-right" transmit control representing the "left-right" movement of a mouse/the person's head linked to a "left-right" receive control representing the "left-right" movement of an image on the visual display simultaneously is indicative of the fact that the output is displayed defining a relationship between said at least one transmit control and said at least one receive control without determining if the type of data input to said animation program is compatible with the type of output from said animation program. (*e.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com). For example, (i) "opening-closing" of a person's mouth is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display or (ii) sounds from talking into a microphone is a transmit control linked to an

"opening-closing" receive control representing the "opening-closing" of an animated mouth on display; because either input can be used, it is using normalized data, and there is live output. Thus, either opening or closing your mouth or talking into a microphone moves the jaw; again, each of those inputs are converted into normalized data so that it can be used to animate the mouth. (*e.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com; *also* https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol; https://helpx.adobe.com/adobe-character-animator/how-to/adobe-character-animator.html).

IV. COUNT II
(PATENT INFRINGEMENT OF UNITED STATES PATENT NO. 6,944,825)

41. The foregoing paragraphs are incorporated by reference.

42. **Direct Infringement.** Upon information and belief, Defendant has been directly infringing claim of the '825 patent in Delaware, and elsewhere in the United States, by making, using, selling, and offering for sale software that satisfies each and every limitation of at least claims 9, 11, and 15, including without limitation the Adobe Character Animator ("Accused Instrumentality"). (*E.g.*, http://www.adobe.com/products/character-animator.html).

43. The Accused Instrumentality is an animator program embodied on a computer-readable medium (*e.g.,* the Accused Instrumentality is software that is stored and operates on personal computers), wherein the output of said animation program is dynamically generated in real time (*e.g.,* the visual display of animations created in the Accused Instrumentality are generated and executed in real-time based on real-time user input). (*E.g.*, *id.*; https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-

animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com).

44. The Accused Instrumentality represents input (*e.g.*, tying user inputs such as mouse inputs, keyboard inputs, or voice inputs) to said animation program with at least one transmit control (*e.g.*, tying user inputs to character animations commands). Different types of inputs (*e.g.*, camera input/mouse & touch input/keyboard input) can be used to control the movement of the animated character on the visual display simultaneously. Thus, "left-right" transmit control representing the "left-right" movement of a mouse/the person's head creates an input function based on the data inputs that generates data that can be used for an output function. (*E.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com). As another example, (i) "opening-closing" of a person's mouth is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display or (ii) sounds from talking into a microphone is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display. (*E.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-

animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol).

45. The Accused Instrumentality represents the output of said animation program (*e.g.*, a visual display of character animation) with at least one receive control (*e.g.*, tying character animations to received user inputs and the animation command linked to the input). For example, the "left-right" transmit control representing the "left-right" movement of a mouse/the person's head linked to a "left-right" receive control representing the "left-right" movement of an image on the visual display simultaneously. (*E.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com). As another example, (i) "opening-closing" of a person's mouth is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display or (ii) sounds from talking into a microphone is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display; because either input can be used, it is using normalized data, and there is live output. Thus, either opening or closing your mouth or talking into a microphone moves the jaw; again, each of those inputs are converted into normalized data so that it can be used to animate the mouth. (*E.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-

behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol).

46. The Accused Instrumentality defines a relationship between said at least one transmit control (*e.g.,* a command that instructs the Accused Instrumentality that a user input has occurred) and said at least one receive control (*e.g.,* a command for a character animation to be output is linked to the particular user input) without determining if the type of data input to said animation program (*e.g.,* user input via mouse, keyboard or voice) is compatible with the type of output from said animation program (*e.g.,* a user can use any input they choose to activate any character animation they choose with no concerns over compatibility). Different types of inputs (e.g., camera input/mouse & touch input/keyboard input) can be used to control the movement of the animated character on the visual display simultaneously. Thus, "left-right" transmit control representing the "left-right" movement of a mouse/the person's head linked to a "left-right" receive control representing the "left-right" movement of an image on the visual display simultaneously is indicative of the fact that the output is displayed defining a relationship between said at least one transmit control and said at least one receive control without determining if the type of data input to said animation program is compatible with the type of output from said animation program. (*E.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-

animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/adobe-character-animator.html; https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol).

47. The Accused Instrumentality dynamically generates the output of said animation program in real-time (*e.g.,* the visual display of character animations is generated in real-time based on real-time user inputs) by changing the input to said animation program (*e.g.,* different user inputs tied to different character animations will cause the dynamic generation of said character animations) represented by said at least one transmit control (*e.g.,* commands that notify the program to execute a character animation) linked to said at least one receive control (*e.g.,* commands that notify the program of a user input has occurred that is linked to a particular character animation). (*E.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/adobe-character-animator.html; https://helpx.adobe.com/adobe-character-animator/using/getting-started.html#webcamcontrol; https://helpx.adobe.com/adobe-character-animator/using/behavior.html#main-pars_header_40).

48. With respect to claim 11, the Accused Instrumentality discloses that the input to said animation program comprises motion input (*e.g.,* movement of head/lips). (*E.g.*,

https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com).

49. With respect to claim 15, the Accused Instrumentality practices a computer-implemented method for changing an animated image in response to input data (*e.g.*, inputs from user via camera & microphone/mouse & touch/keyboard) from an input device (*e.g.*, camera & microphone/mouse & touch/keyboard). The Accused Instrumentality is an animator program embodied on a computer-readable medium (*e.g.*, the Accused Instrumentality is software that is stored and operates on personal computers), wherein the output of said animation program is dynamically generated in real time (*e.g.*, the visual display of animations created in the Accused Instrumentality are generated and executed in real-time based on real-time user input). (*E.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com; http://www.adobe.com/products/character-animator.html).

50. The Accused Instrumentality receives input data from the input device; translating the input data from the input device into normalized data. Upon information and belief, the

Accused Instrumentality normalizes data so they can have substantially different inputs that can still result in the same type of output. Different types of inputs (*e.g.*, camera input/mouse & touch input/keyboard input) can be used to control the movement of the animated character on the visual display simultaneously. Thus, "left-right" transmit control representing the "left-right" movement of a mouse/the person's head linked to a "left-right" receive control representing the "left-right" movement of an animated character on the visual display simultaneously is indicative of the fact that the output is displayed defining a relationship between said at least one transmit control and said at least one receive control without determining if the type of data input to said animation program is compatible with the type of output from said animation program. (*E.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com; http://www.adobe.com/products/character-animator.html).

51. The Accused Instrumentality translates the normalized data into output data associated with the animated image, wherein translating the input data into normalized data and translating the normalized data into output data is based on a relationship defined between the input data and the output data, and wherein said relationship is defined without determining if the type of input data is compatible with the type of output data. (*E.g.*, https://helpx.adobe.com/adobe-character-animator/how-to/use-triggers-to-control-animation-behaviors.html?playlist=/ccx/v1/collection/product/adobe-character-

animator/segment/designer/explevel/beginner/applaunch/orientation/collection.ccx.js?ref=helpx.adobe.com; https://helpx.adobe.com/adobe-character-animator/how-to/create-puppet-animation.html?playlist=/ccx/v1/collection/product/adobe-character-animator/topics/animation/collection.ccx.js?ref=helpx.adobe.com; http://www.adobe.com/products/character-animator.html). For example, Different types of inputs (e.g., camera input/mouse & touch input/keyboard input) can be used to control the movement of the animated character on the visual display simultaneously. Thus, "left-right" transmit control representing the "left-right" movement of a mouse/the person's head linked to a "left-right" receive control representing the "left-right" movement of an animated character on the visual display simultaneously is indicative of the fact that the output is displayed defining a relationship between said at least one transmit control and said at least one receive control without determining if the type of data input to said animation program is compatible with the type of output from said animation program. (*Id.*). As another example, (i) "opening-closing" of a person's mouth is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display or (ii) sounds from talking into a microphone is a transmit control linked to an "opening-closing" receive control representing the "opening-closing" of an animated mouth on display; because either input can be used, it is using normalized data, and there is live output. Thus, either opening or closing your mouth or talking into a microphone moves the jaw; again, each of those inputs are converted into normalized data so that it can be used to animate the mouth. (*Id.*).

52. The Accused Instrumentality causes the animated image to change in real-time in reaction to the input data that is received from the input device and that is translated into the output data associated with the animated image. The Accused Instrumentality is an animator

program embodied on a computer-readable medium (*e.g.*, the Accused Instrumentality is software that is stored and operates on personal computers), wherein the output of said animation program is dynamically generated in real time (*e.g.*, the visual display of animations created in the Accused Instrumentality are generated and executed in real-time based on real-time user input). (*Id.*).

53. Plaintiff has been damaged as a result of Defendant's infringing conduct. Defendant is thus liable to Plaintiff for damages in an amount that adequately compensates Plaintiff for Defendant's infringement of the '864 patent and '825 patent, *i.e.*, in an amount that by law cannot be less than would constitute a reasonable royalty for the use of the patented technology, together with interest and costs as fixed by this Court under 35 U.S.C. § 284.

54. On information and belief, Defendant has had at least constructive notice of the '864 patent and '825 patent by operation of law, and there are no marking requirements that have not been complied with.

IV. **JURY DEMAND**

Plaintiff, under Rule 38 of the Federal Rules of Civil Procedure, requests a trial by jury of any issues so triable by right.

V. **PRAYER FOR RELIEF**

WHEREFORE, Plaintiff respectfully requests that the Court find in its favor and against Defendant, and that the Court grant Plaintiff the following relief:

a. Judgment that one or more claims of United States Patent No. 6,331,864 have been infringed, either literally and/or under the doctrine of equivalents, by Defendant;

b. Judgment that one or more claims of United States Patent No. 6,944,825 have been infringed, either literally and/or under the doctrine of equivalents, by Defendant;

c. Judgment that Defendant account for and pay to Plaintiff all damages to and costs incurred by Plaintiff because of Defendant's infringing activities and other conduct complained of herein;

d. That Plaintiff be granted pre-judgment and post-judgment interest on the damages caused by Defendant's infringing activities and other conduct complained of herein; and

e. That Plaintiff be granted such other and further relief as the Court may deem just and proper under the circumstances.

July 18, 2018

OF COUNSEL:

David R. Bennett
Direction IP Law
P.O. Box 14184
Chicago, IL 60614-0184
(312) 291-1667
dbennett@directionip.com

STAMOULIS & WEINBLATT LLC

/s/ Stamatios Stamoulis
Stamatios Stamoulis
Two Fox Point Centre
6 Denny Road, Suite 307
Wilmington, DE 19809
(302) 999-1540
stamoulis@swdelaw.com

Attorneys for Plaintiff Wiremed Tech LLC