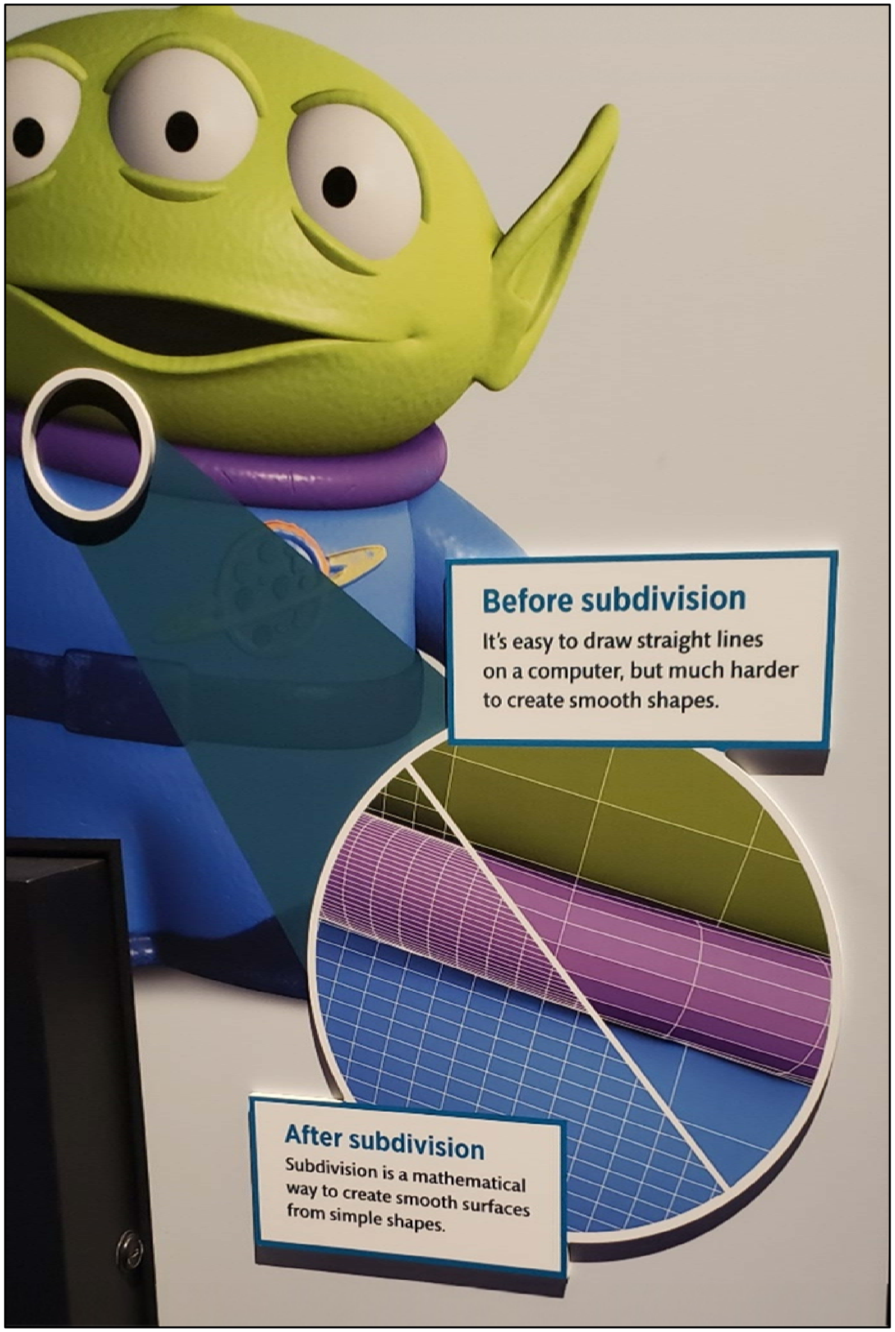
The Science of Pixar

At the Oregon Museum of Science and Industry (OMSI)

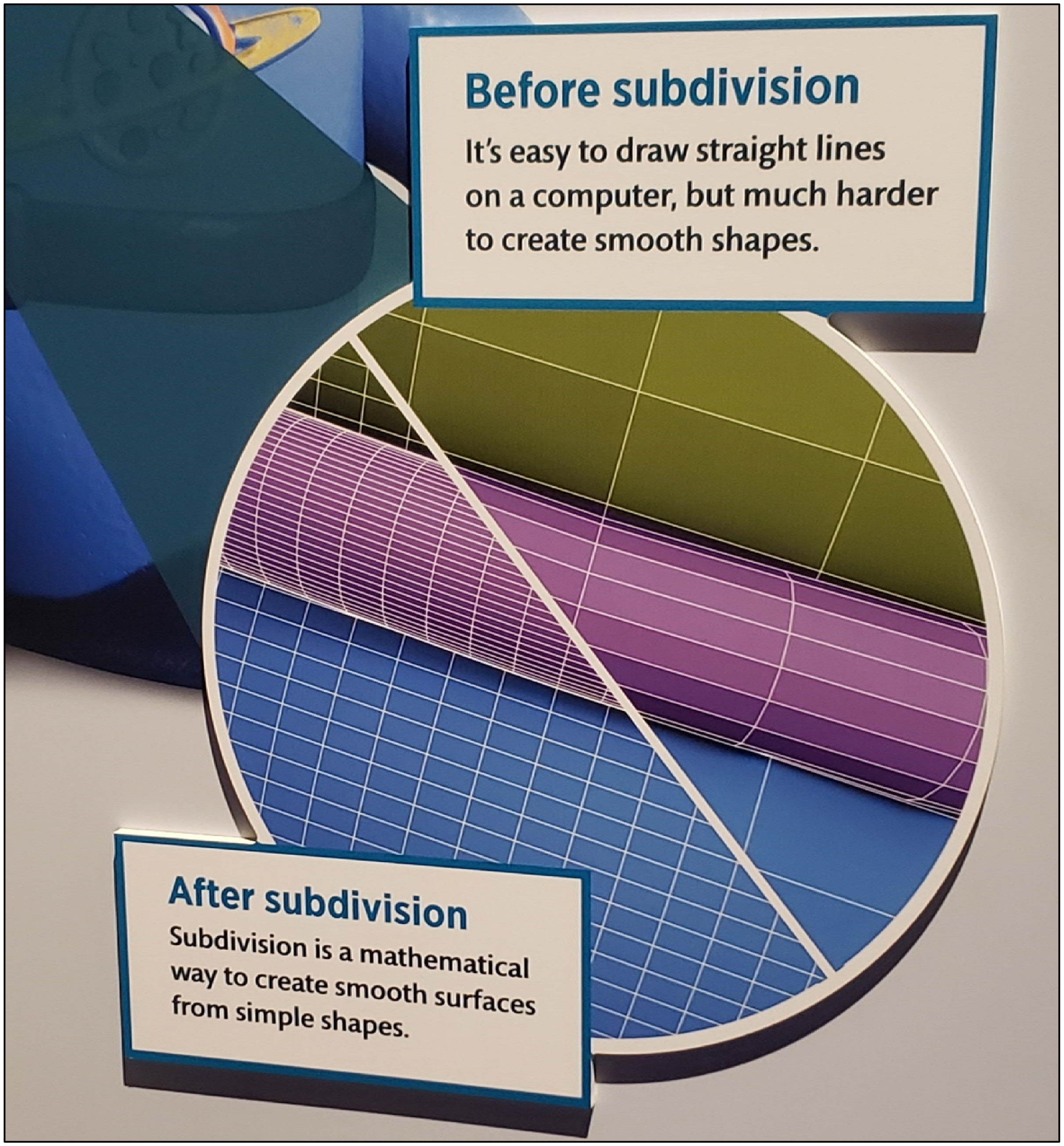


Before subdivision

It's easy to draw straight lines on a computer, but much harder to create smooth shapes.

After subdivision

Subdivision is a mathematical way to create smooth surfaces from simple shapes.

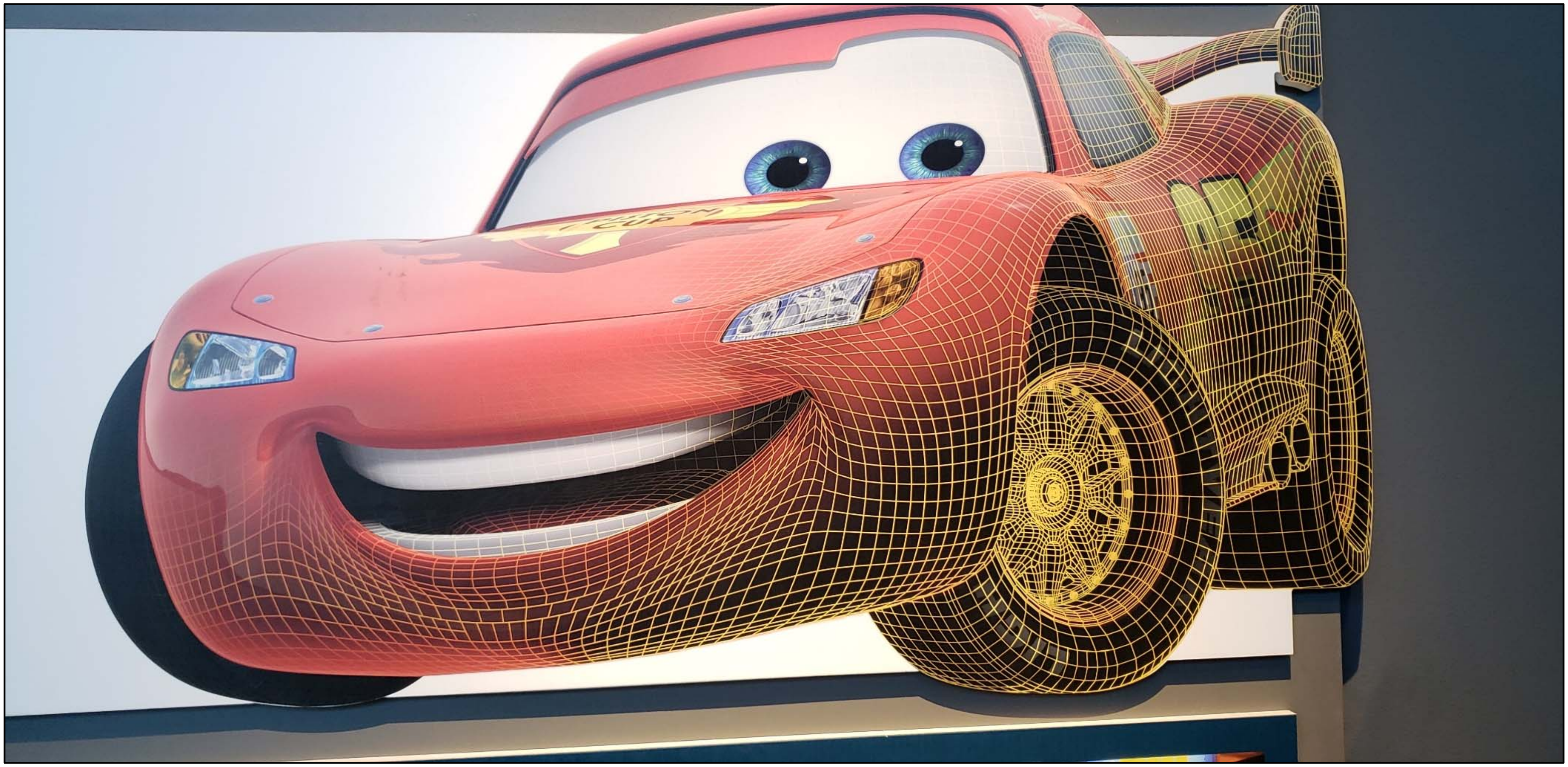


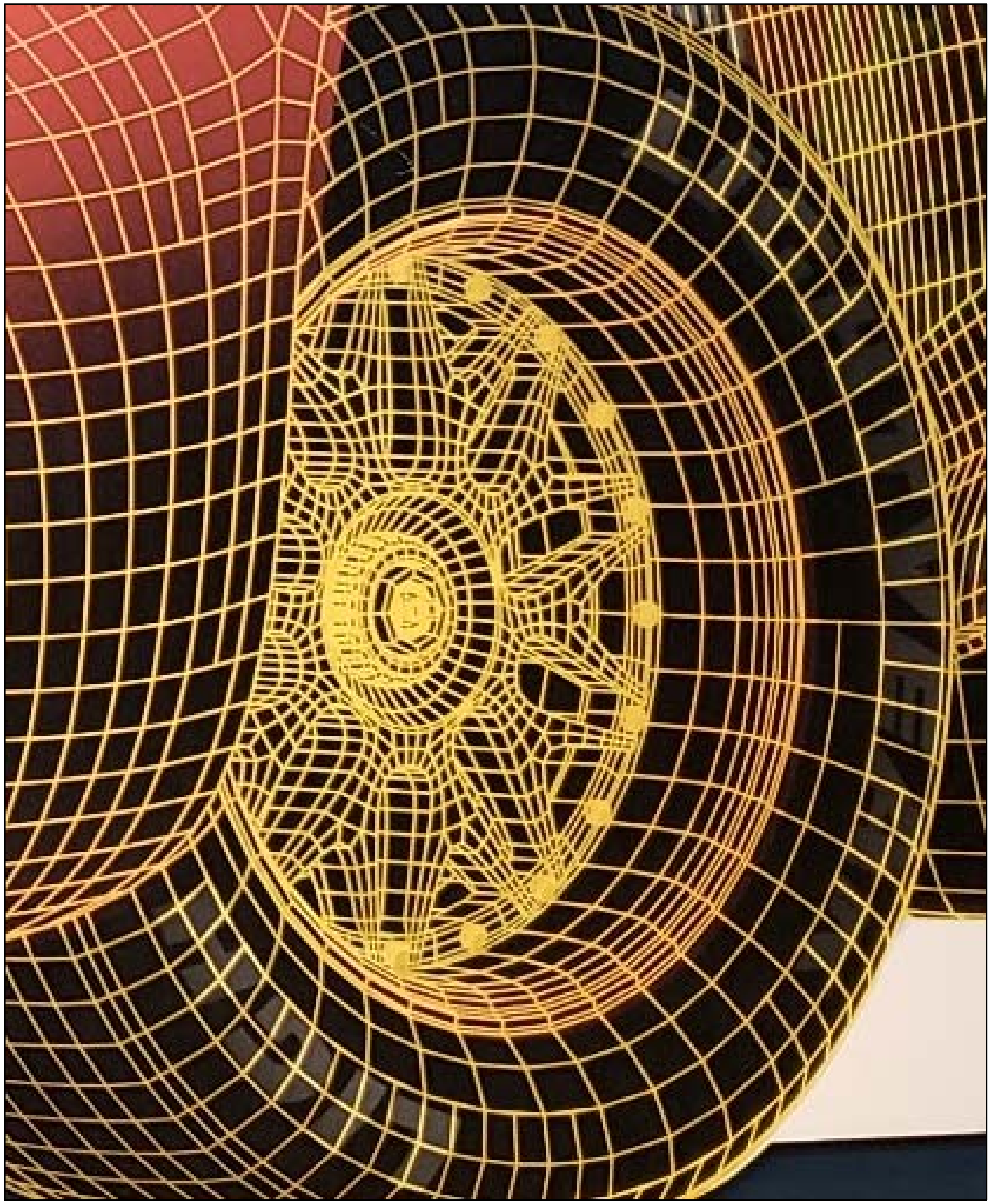
Before subdivision

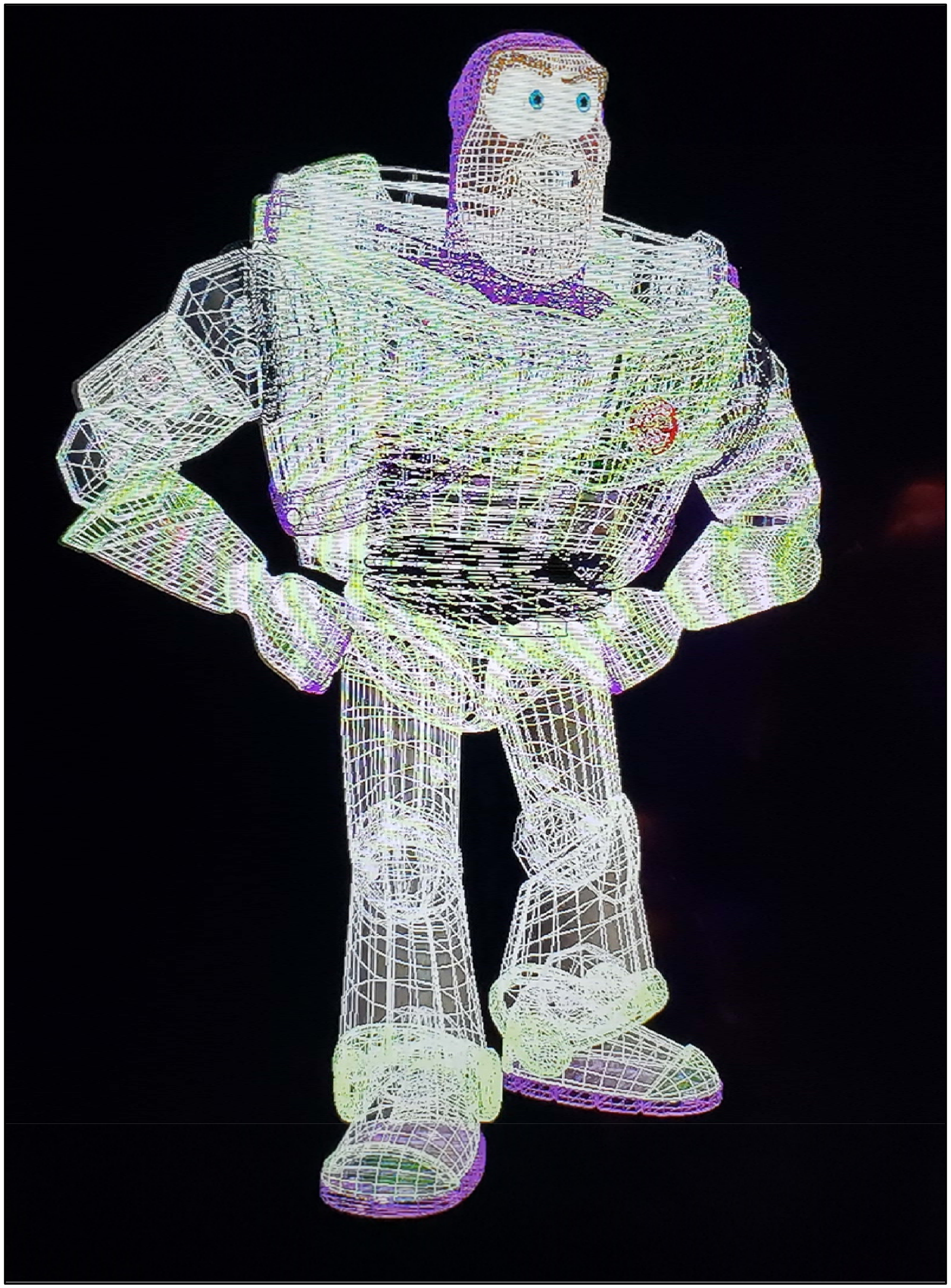
It's easy to draw straight lines on a computer, but much harder to create smooth shapes.

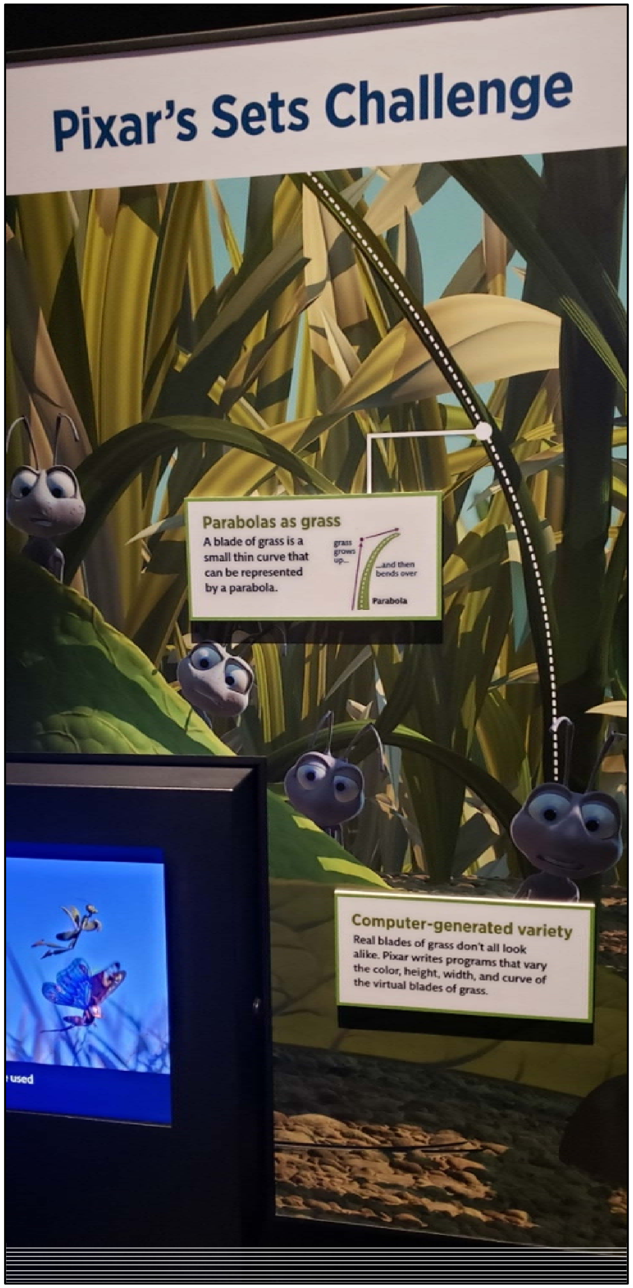
After subdivision

Subdivision is a mathematical way to create smooth surfaces from simple shapes.









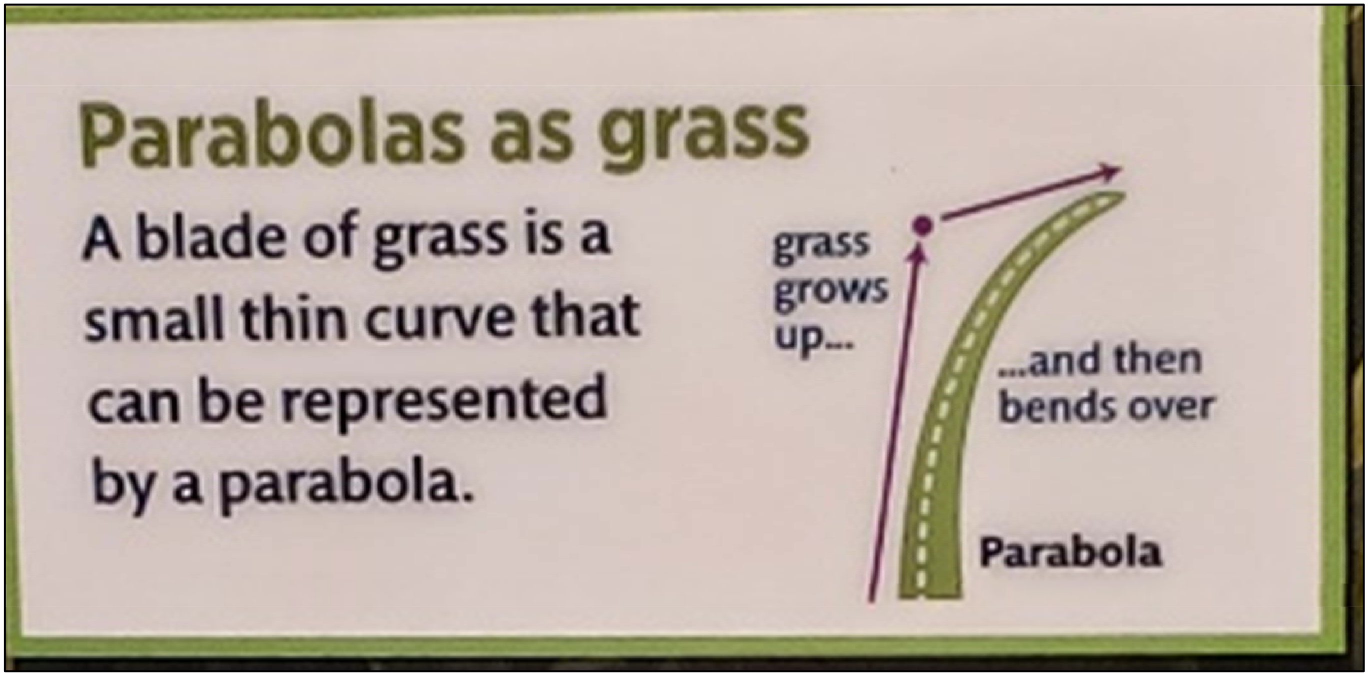
Pixar's Sets Challenge

Parabolas as grass

A blade of grass is a small thin curve that can be represented by a parabola.

Computer-generated variety

Real blades of grass don't all look alike. Pixar writes programs that vary the color, height, width, and curve of the virtual blades of grass.



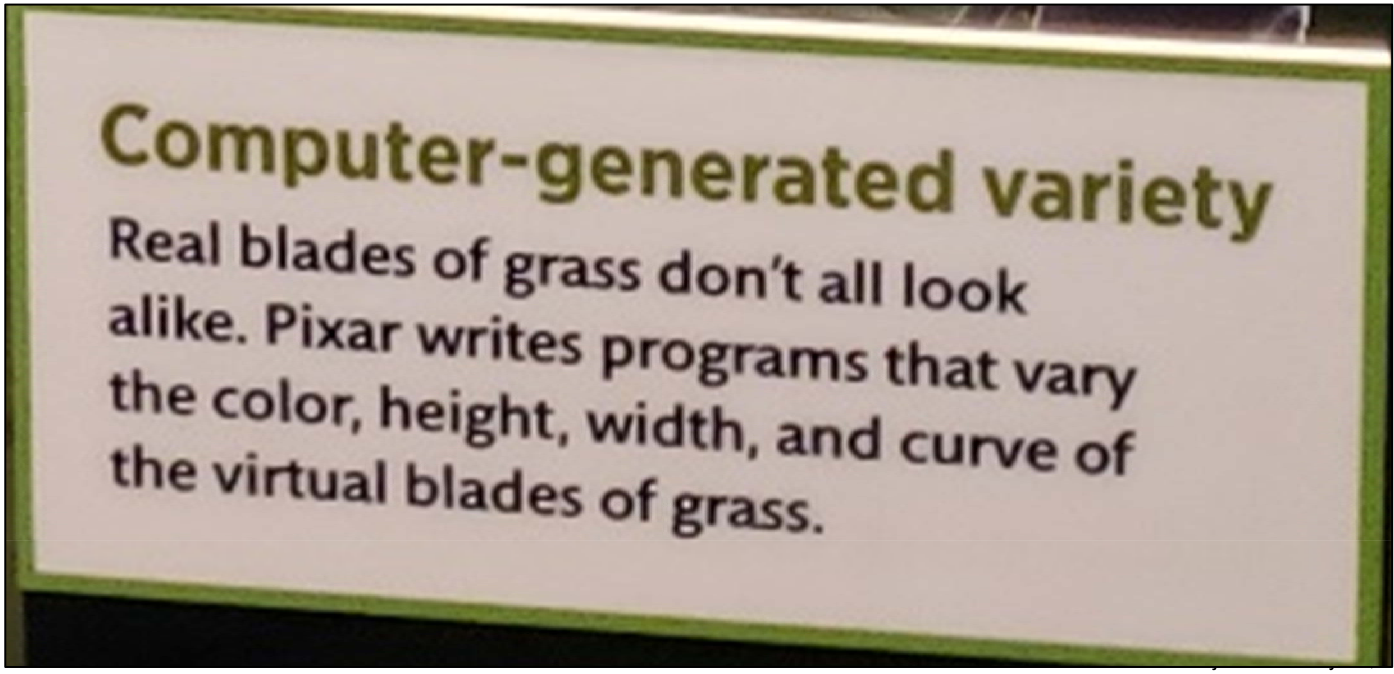
Parabolas as grass

A blade of grass is a small thin curve that can be represented by a parabola.

grass grows up...

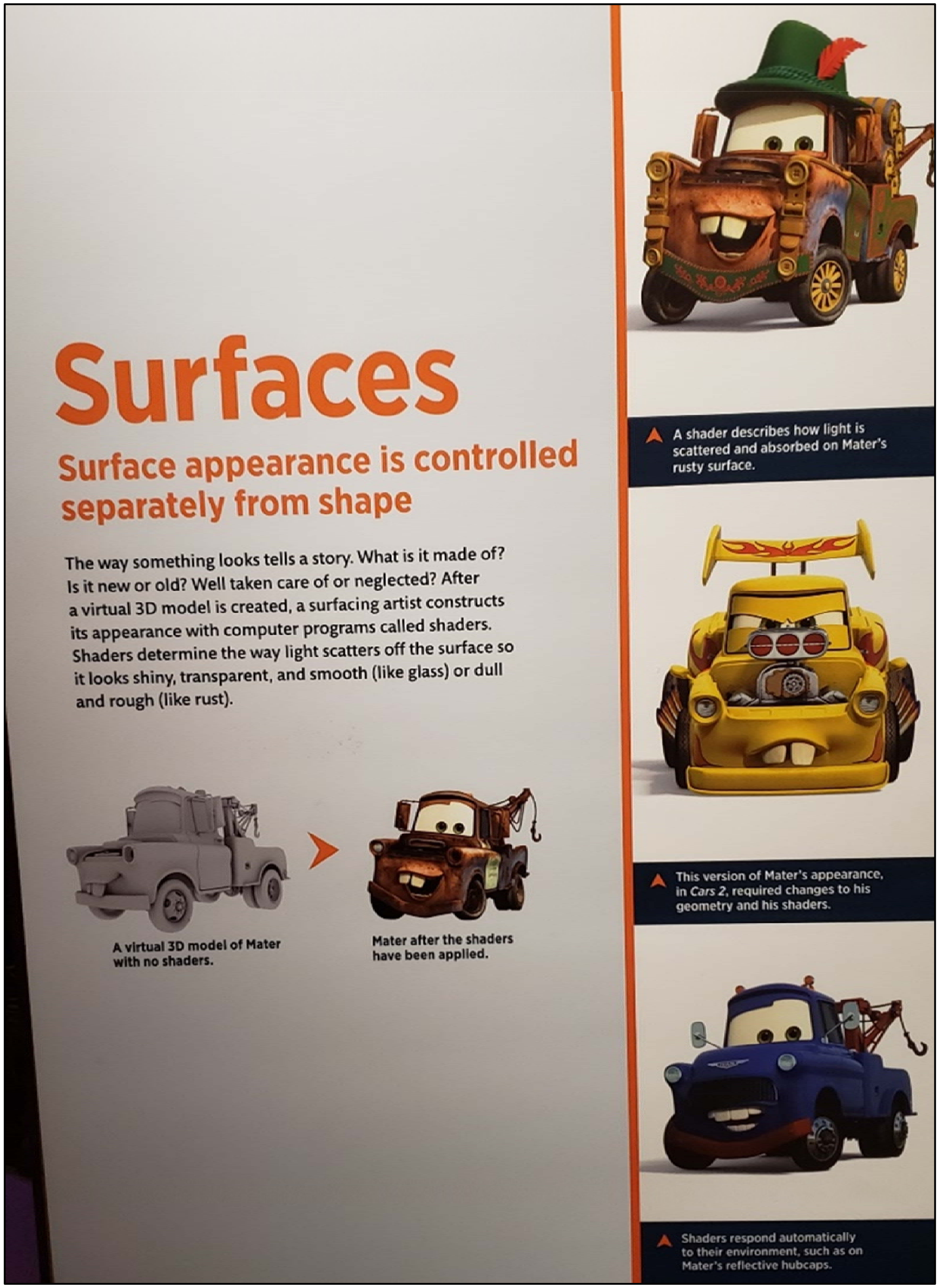
...and then bends over

Parabola



Computer-generated variety

Real blades of grass don't all look alike. Pixar writes programs that vary the color, height, width, and curve of the virtual blades of grass.



Surfaces

Surface appearance is controlled separately from shape

The way something looks tells a story. What is it made of? Is it new or old? Well taken care of or neglected? After a virtual 3D model is created, a surfacing artist constructs its appearance with computer programs called shaders. Shaders determine the way light scatters off the surface so it looks shiny, transparent, and smooth (like glass) or dull and rough (like rust).

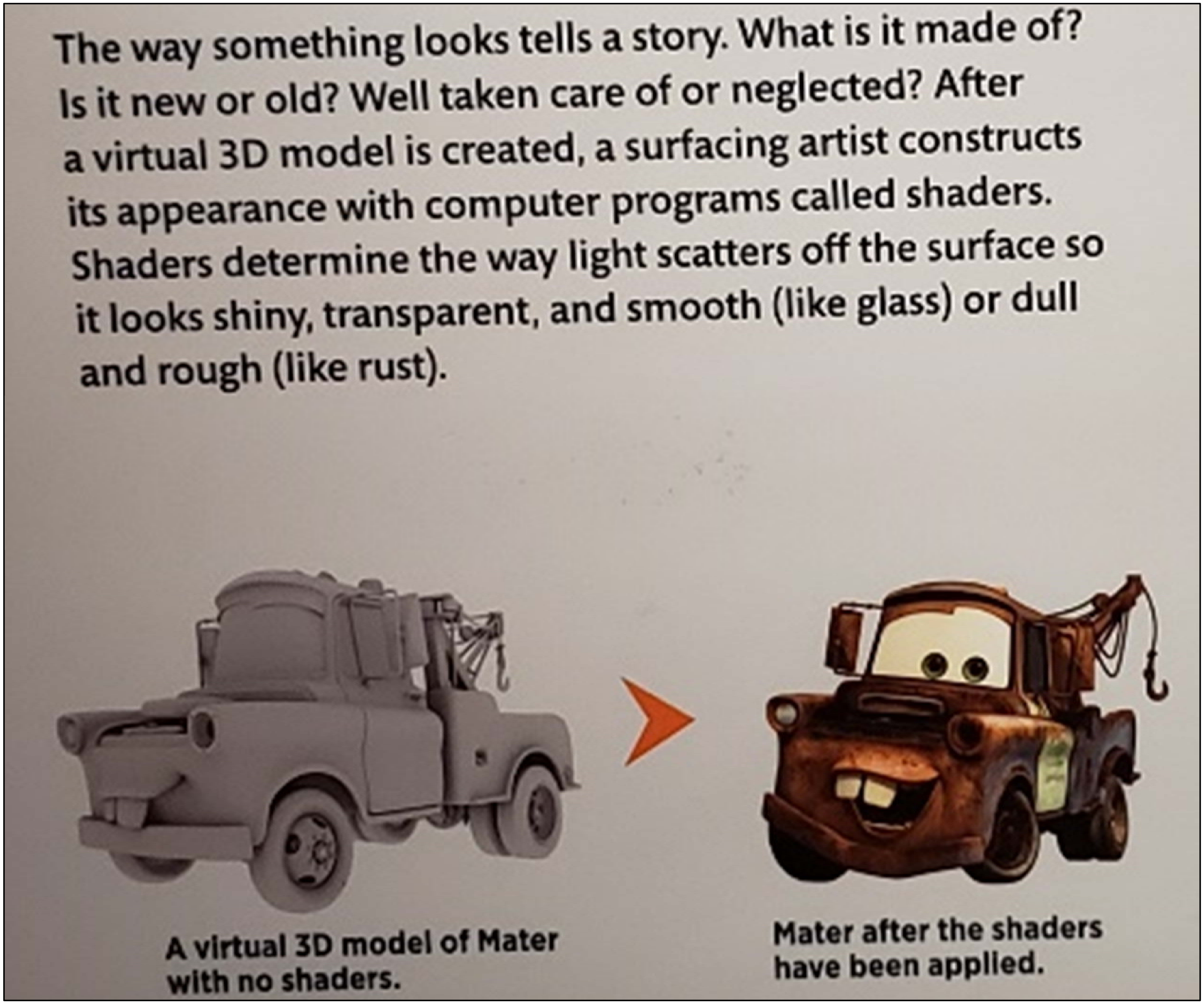
A virtual 3D model of Mater with no shaders.

Mater after the shaders have been applied.

A shader describes how light is scattered and absorbed on Mater's rusty surface.

This version of Mater's appearance, in *Cars 2*, required changes to his geometry and his shaders.

Shaders respond automatically to their environment, such as on Mater's reflective hubcaps.



The way something looks tells a story. What is it made of? Is it new or old? Well taken care of or neglected? After a virtual 3D model is created, a surfacing artist constructs its appearance with computer programs called shaders. Shaders determine the way light scatters off the surface so it looks shiny, transparent, and smooth (like glass) or dull and rough (like rust).

A virtual 3D model of Mater with no shaders.

Mater after the shaders have been applied.

way copper reflects light.

Shaders

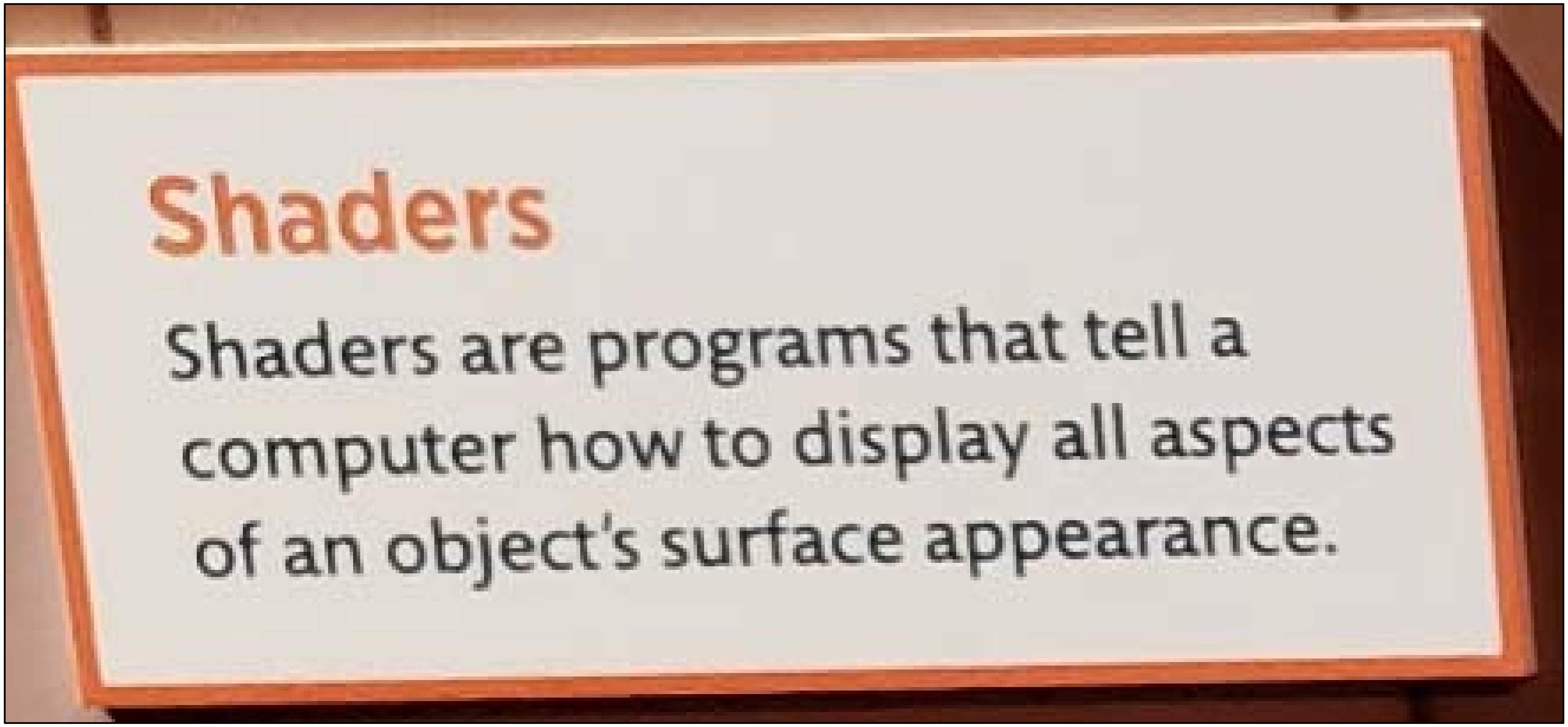
Shaders are programs that tell a computer how to display all aspects of an object's surface appearance.

A worn pot handle

A shader created a bump map that instructs light to bounce unevenly off the handle's surface.

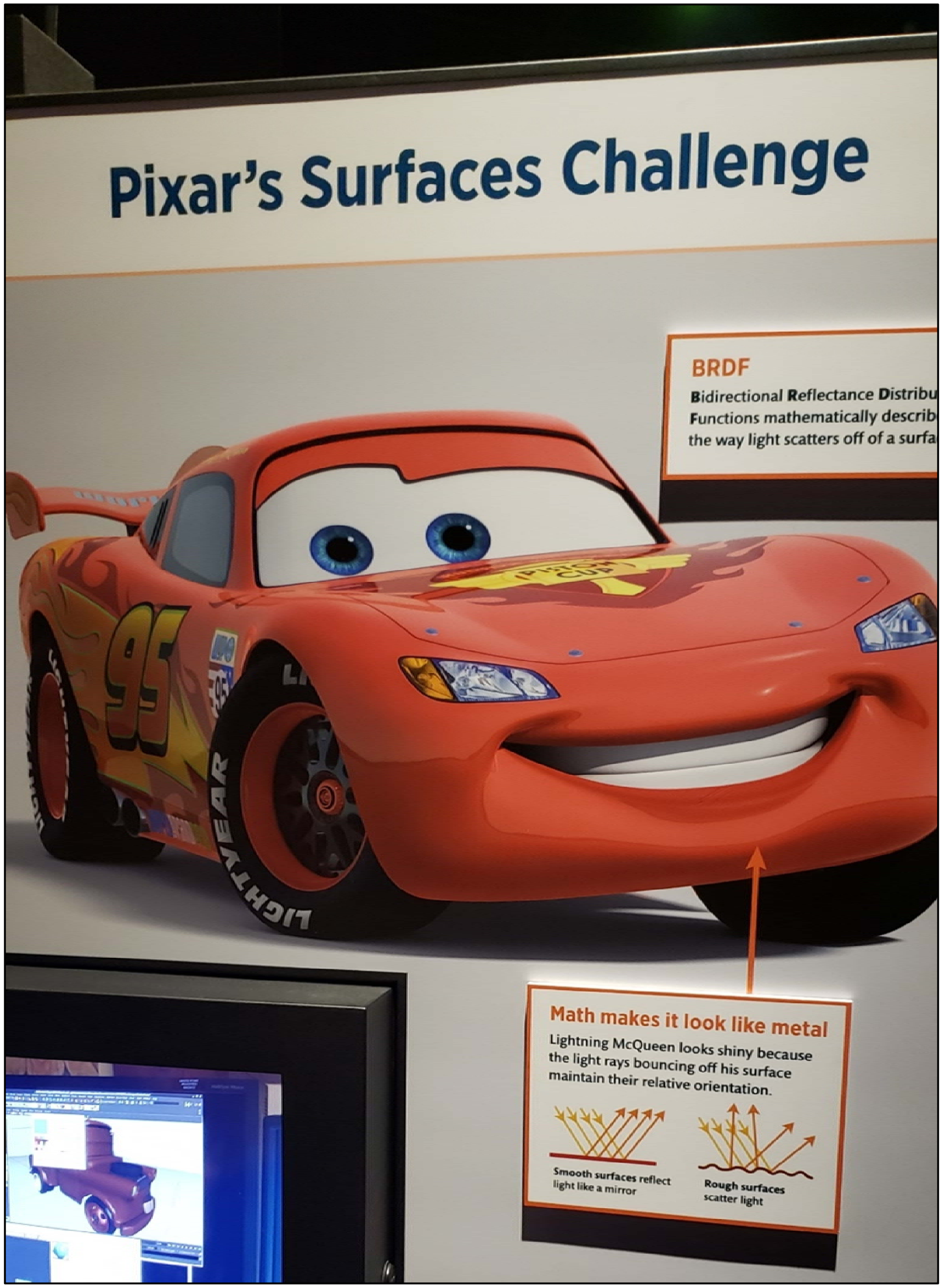
Red onions

A surface pattern shader gives these onions a mottled appearance.



Shaders

Shaders are programs that tell a computer how to display all aspects of an object's surface appearance.



Pixar's Surfaces Challenge

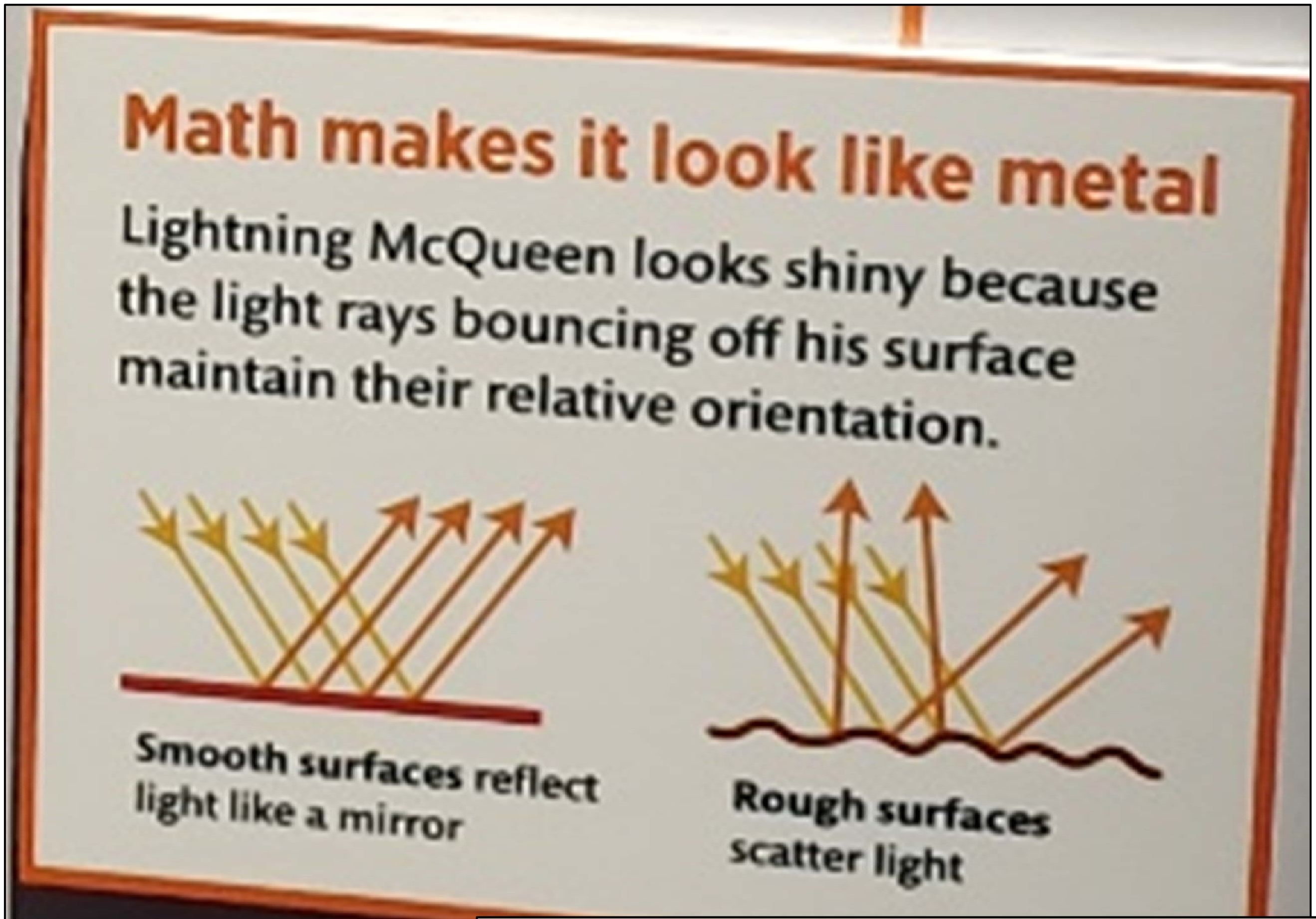
BRDF

Bidirectional **R**eflectance **D**istribu
Functions mathematically describ
the way light scatters off of a surfa

Math makes it look like metal

Lightning McQueen looks shiny because the light rays bouncing off his surface maintain their relative orientation.

Smooth surfaces reflect light like a mirror
Rough surfaces scatter light
95
PISTON CUP
LIGHTYEAR

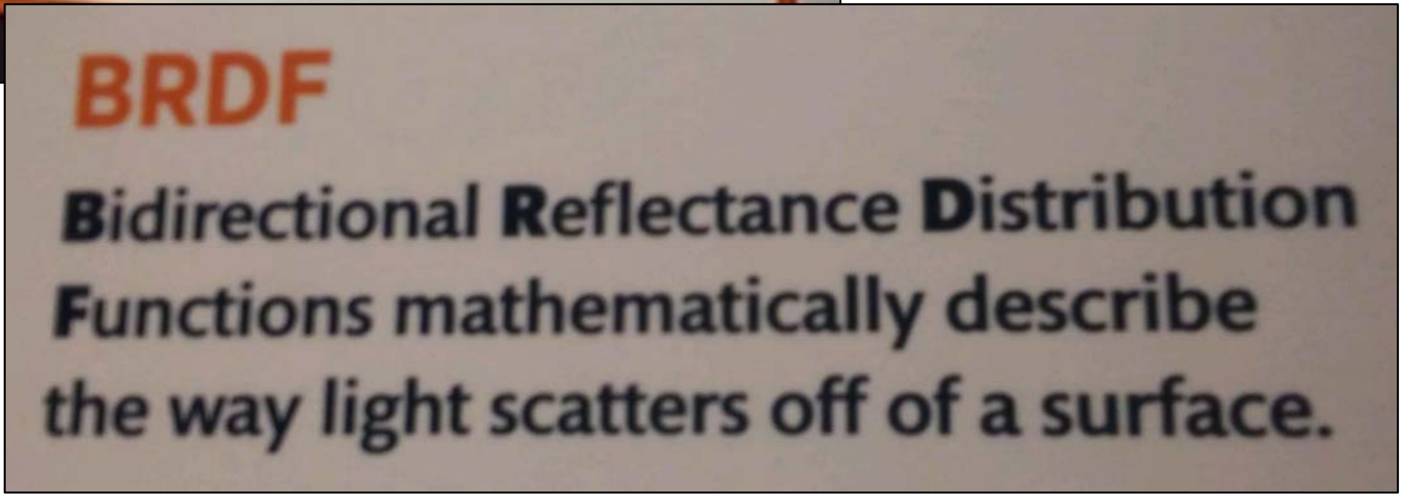


Math makes it look like metal

Lightning McQueen looks shiny because the light rays bouncing off his surface maintain their relative orientation.

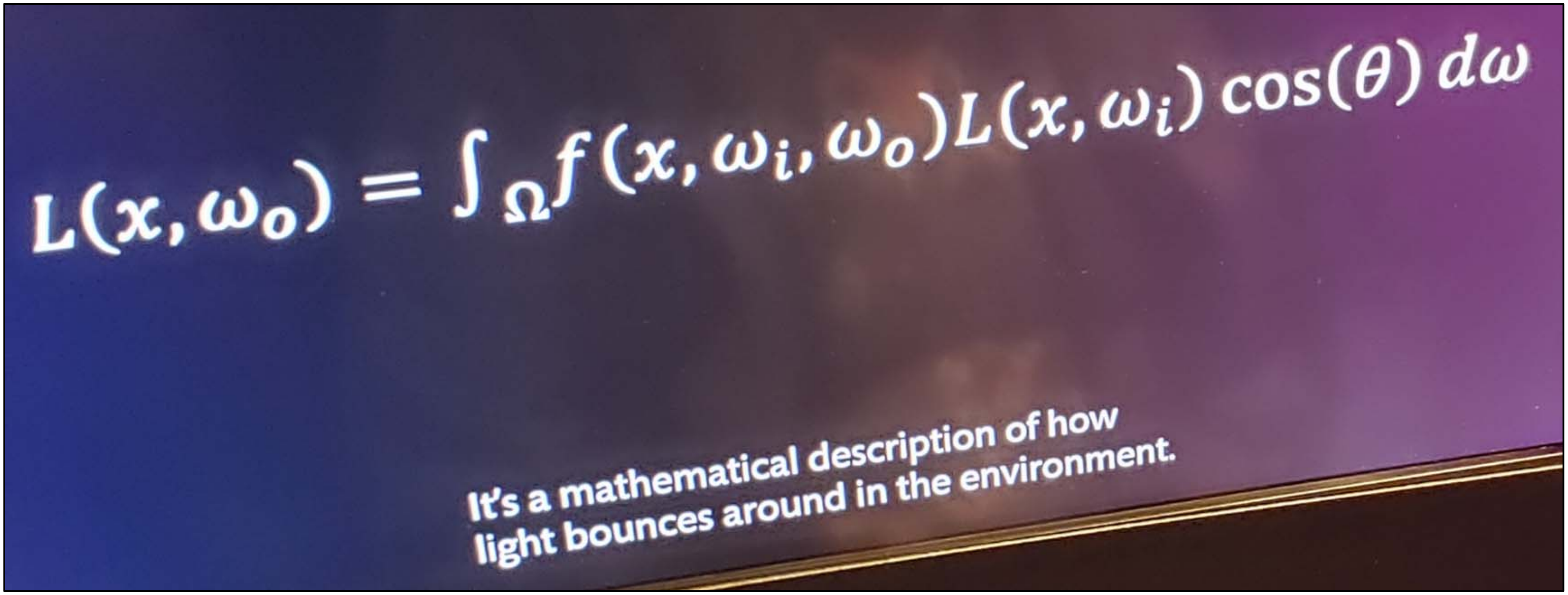
Smooth surfaces reflect light like a mirror

Rough surfaces scatter light



BRDF

Bidirectional **R**eflectance **D**istribution **F**unctions mathematically describe the way light scatters off of a surface.

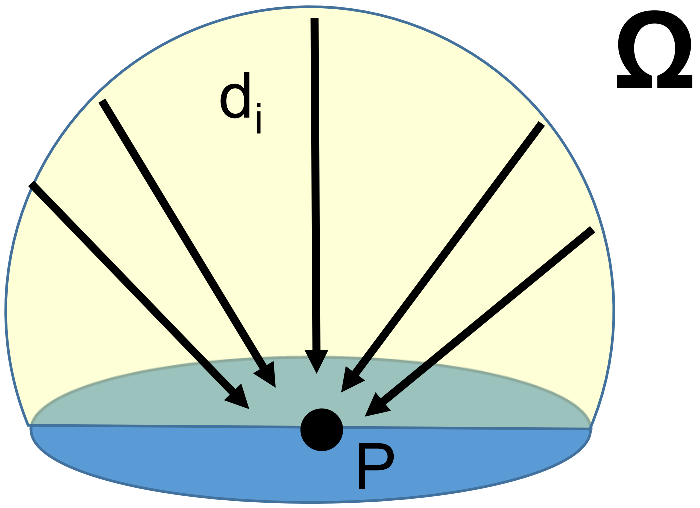


$$L(x,\omega_o) = \int_{\Omega} f(x,\omega_i,\omega_o)L(x,\omega_i)\cos(\theta)\,d\omega$$

It’s a mathematical description of how light bounces around in the environment.

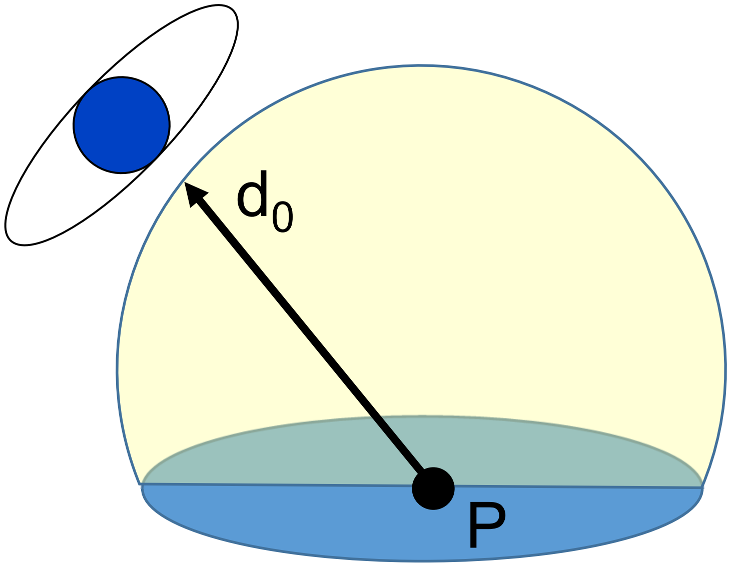
The Rendering Equation

Light Arriving

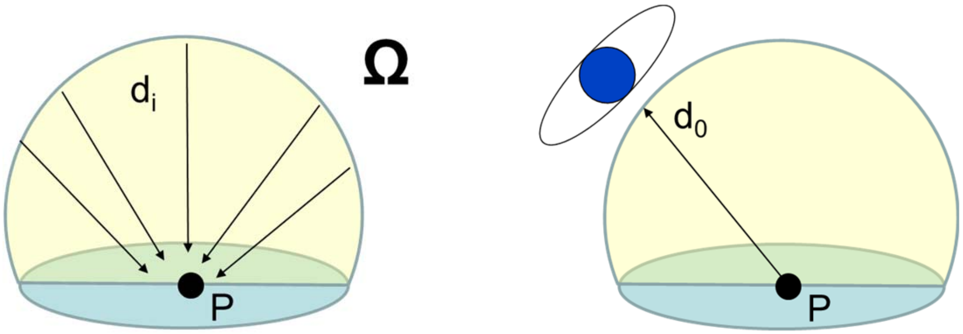


$$L(P,d_0,\lambda) = E(P,d_0,\lambda) + \int_{\Omega} L(P,d_i,\lambda) f(\lambda,d_i,d_0)(d_i \cdot \hat{n}) d\Omega$$

Light Departing



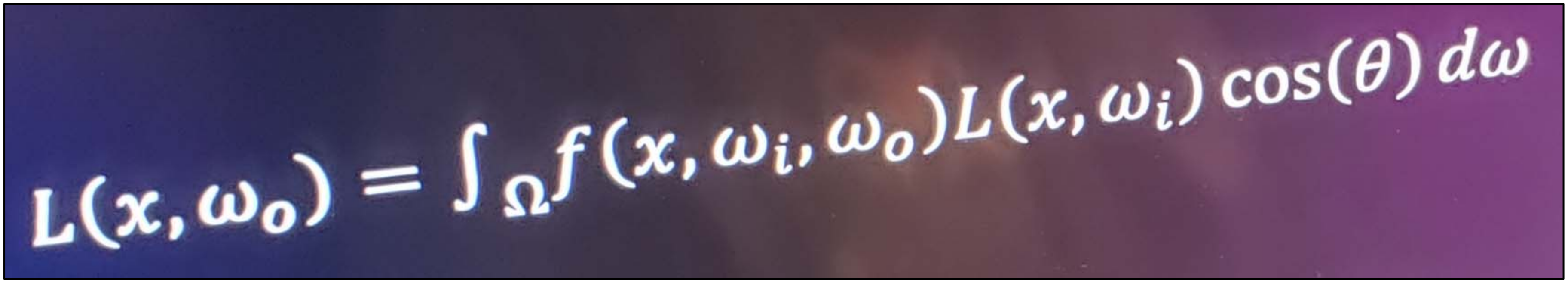
The Rendering Equation



$$B(x, d_0, \lambda) = E(P, d_0, \lambda) + \int_{\Omega} B(x, d_i, \lambda) f(x, \lambda, d_i, d_0)(d_i \cdot \hat{n}) d\Omega$$

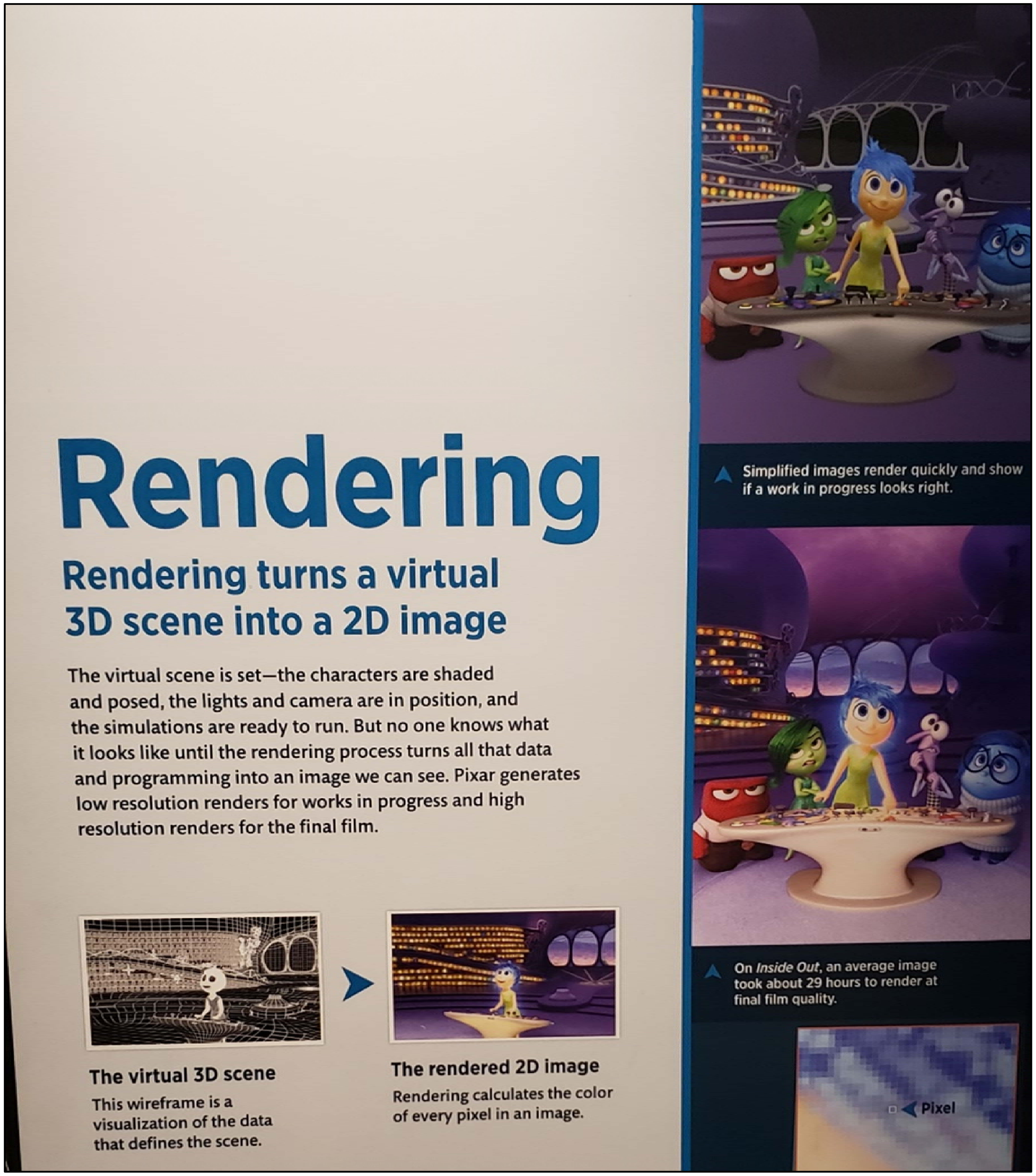
In plain language, this is a simultaneous-equation energy balance:

“The light shining from the point P is the reflection of the incoming light directed to the point P from all of the other points in the scene.”



$$L(x, \omega_o) = \int_\Omega f(x, \omega_i, \omega_o) L(x, \omega_i) \cos(\theta)\, d\omega$$

$$L(x, d_0, \lambda) = E(P, d_0, \lambda) + \int_\Omega L(x, d_i, \lambda) f(x, \lambda, d_i, d_0)(d_i \cdot \hat{n}) d\Omega$$



Rendering

Rendering turns a virtual 3D scene into a 2D image

The virtual scene is set—the characters are shaded and posed, the lights and camera are in position, and the simulations are ready to run. But no one knows what it looks like until the rendering process turns all that data and programming into an image we can see. Pixar generates low resolution renders for works in progress and high resolution renders for the final film.

The virtual 3D scene

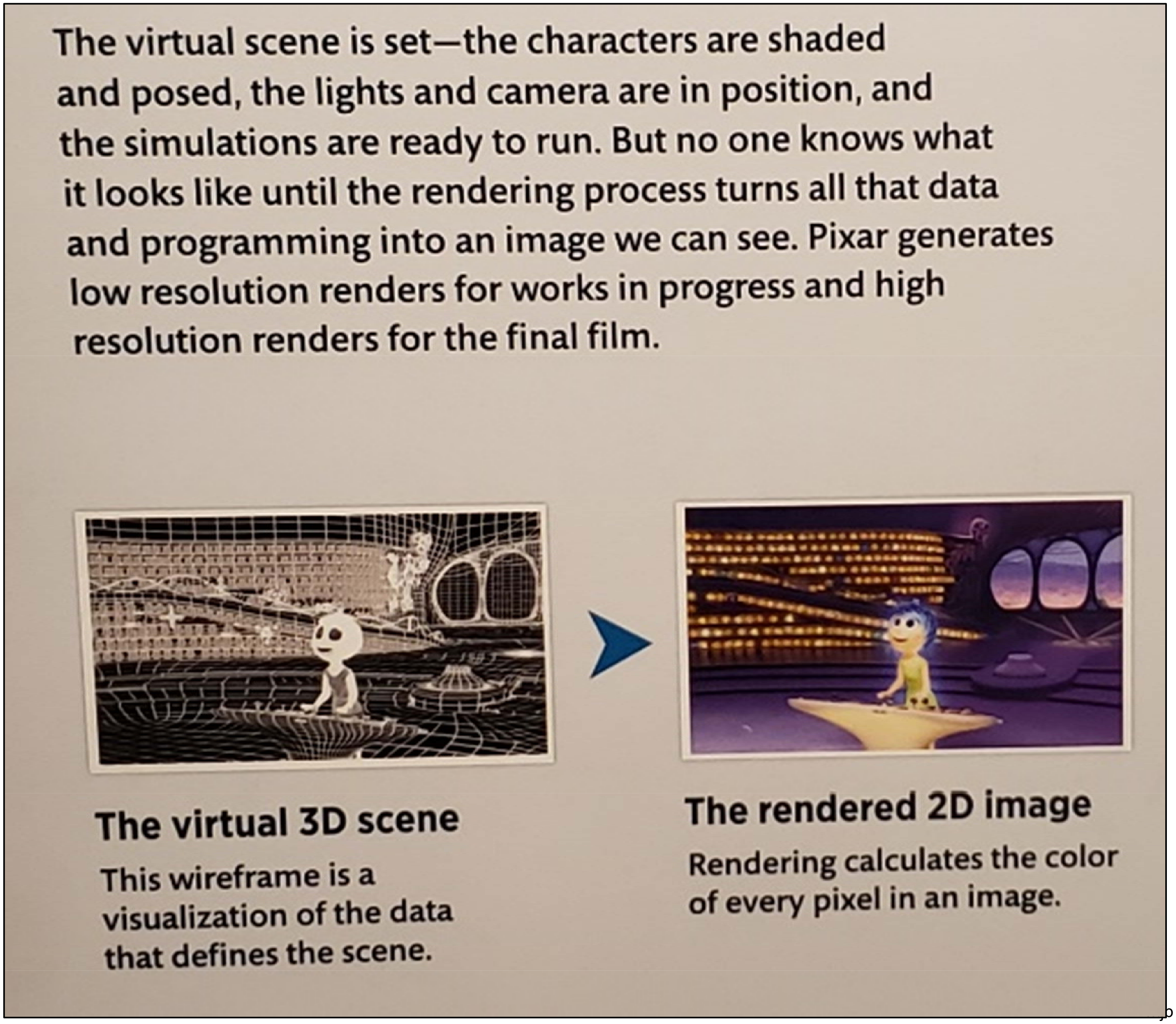
This wireframe is a visualization of the data that defines the scene.

The rendered 2D image

Rendering calculates the color of every pixel in an image.

▲ Simplified images render quickly and show if a work in progress looks right.

▲ On *Inside Out*, an average image took about 29 hours to render at final film quality.



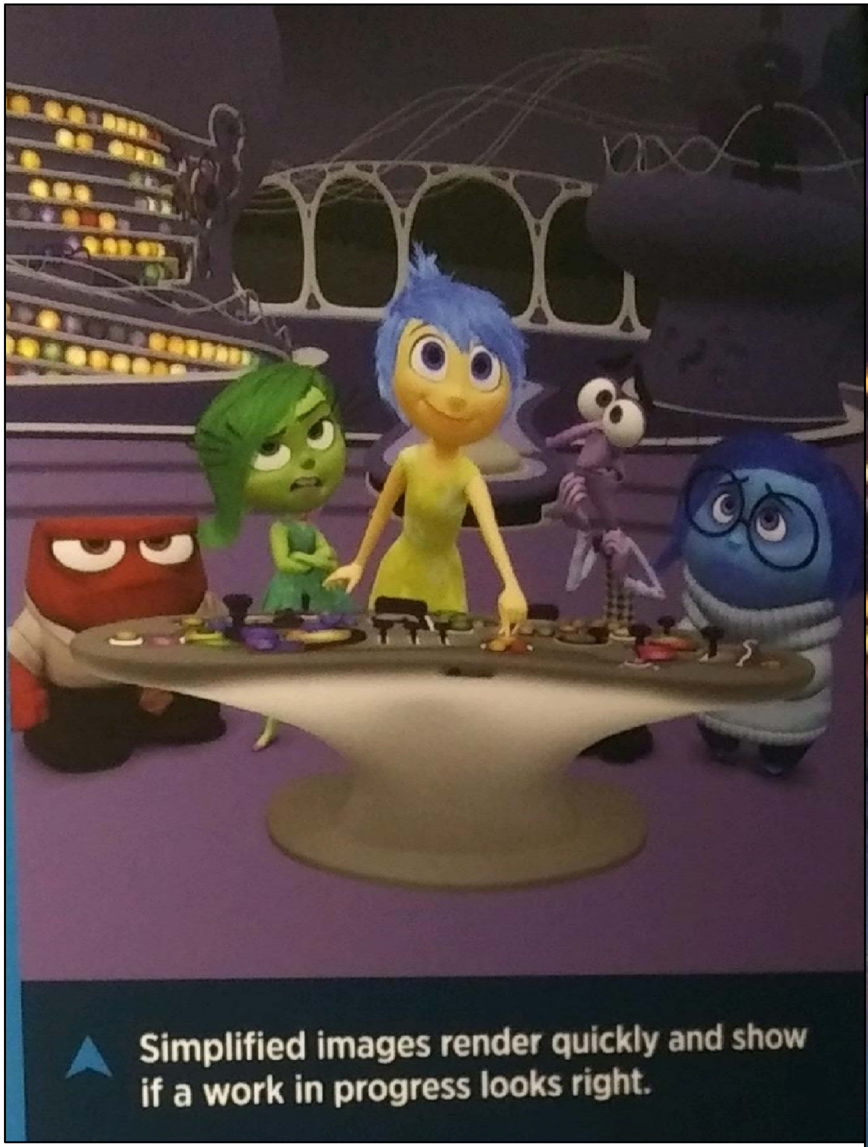
The virtual scene is set—the characters are shaded and posed, the lights and camera are in position, and the simulations are ready to run. But no one knows what it looks like until the rendering process turns all that data and programming into an image we can see. Pixar generates low resolution renders for works in progress and high resolution renders for the final film.

The virtual 3D scene

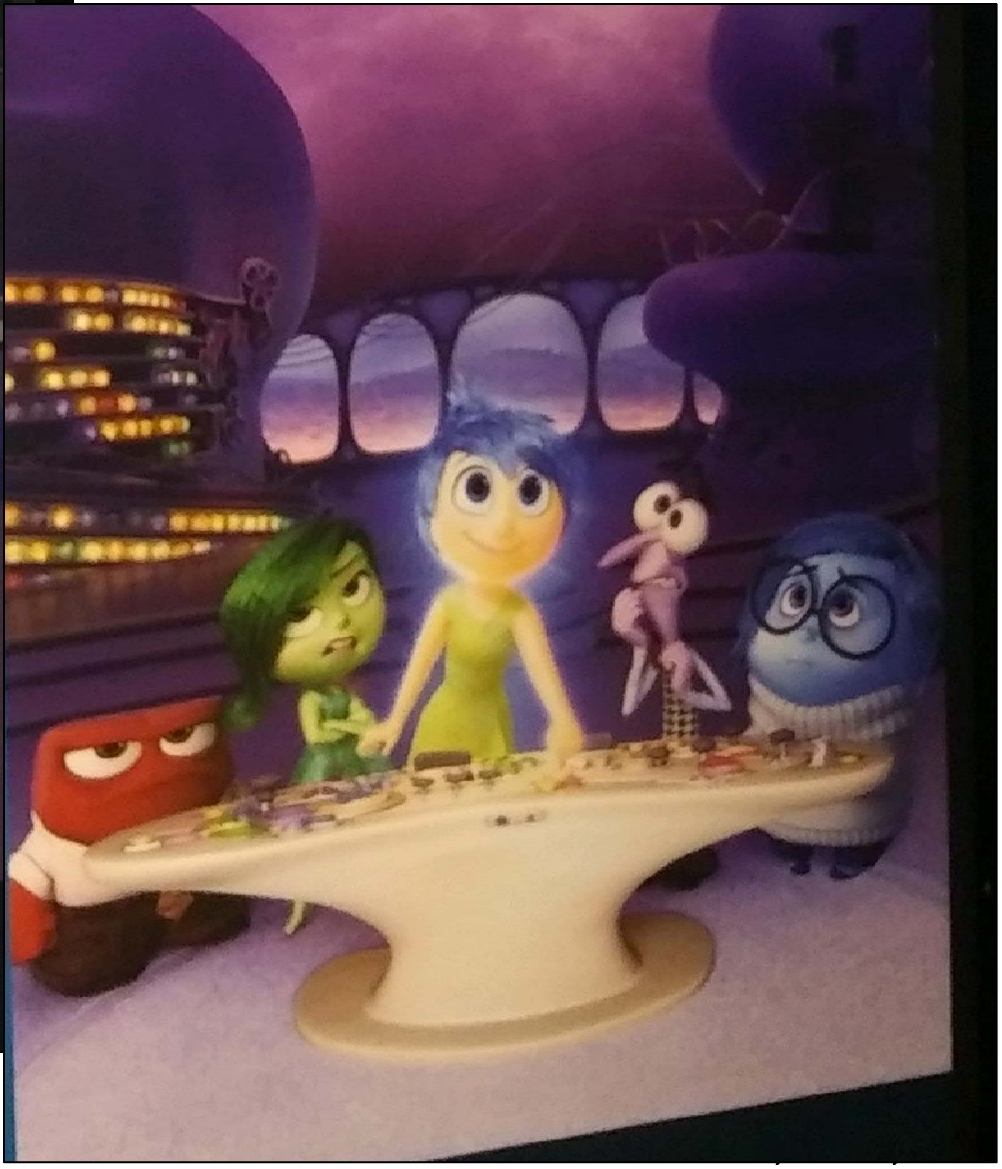
This wireframe is a visualization of the data that defines the scene.

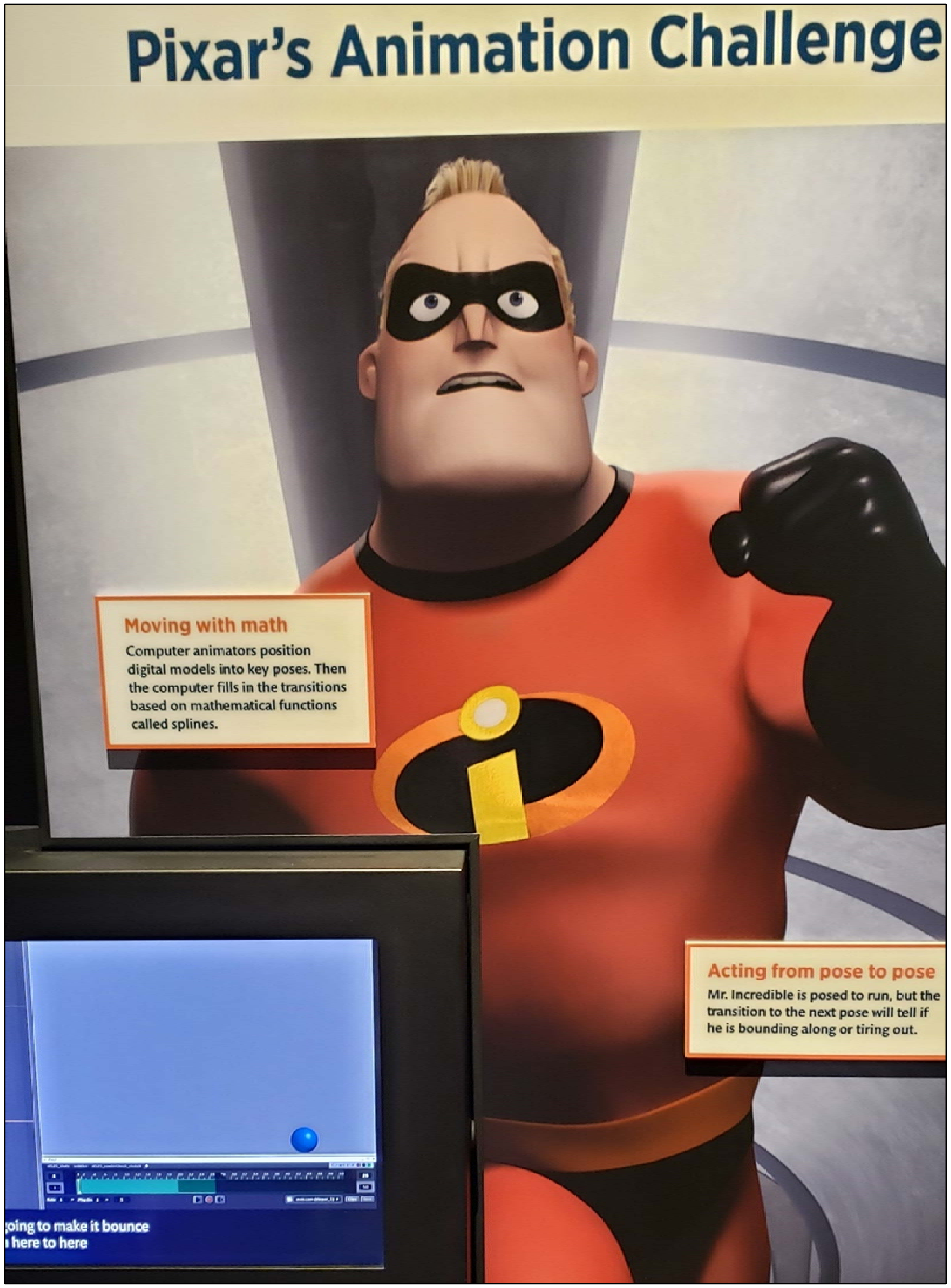
The rendered 2D image

Rendering calculates the color of every pixel in an image.



Simplified images render quickly and show if a work in progress looks right.



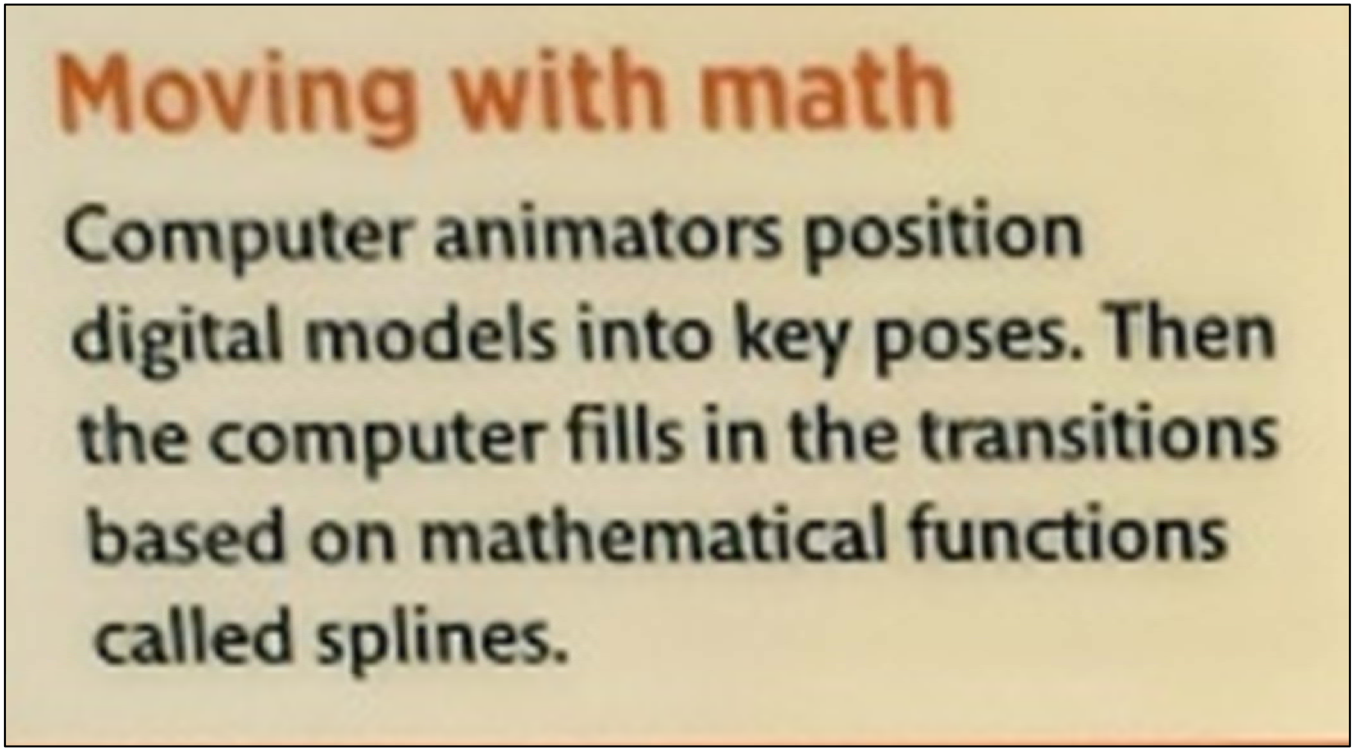


Moving with math

Computer animators position digital models into key poses. Then the computer fills in the transitions based on mathematical functions called splines.

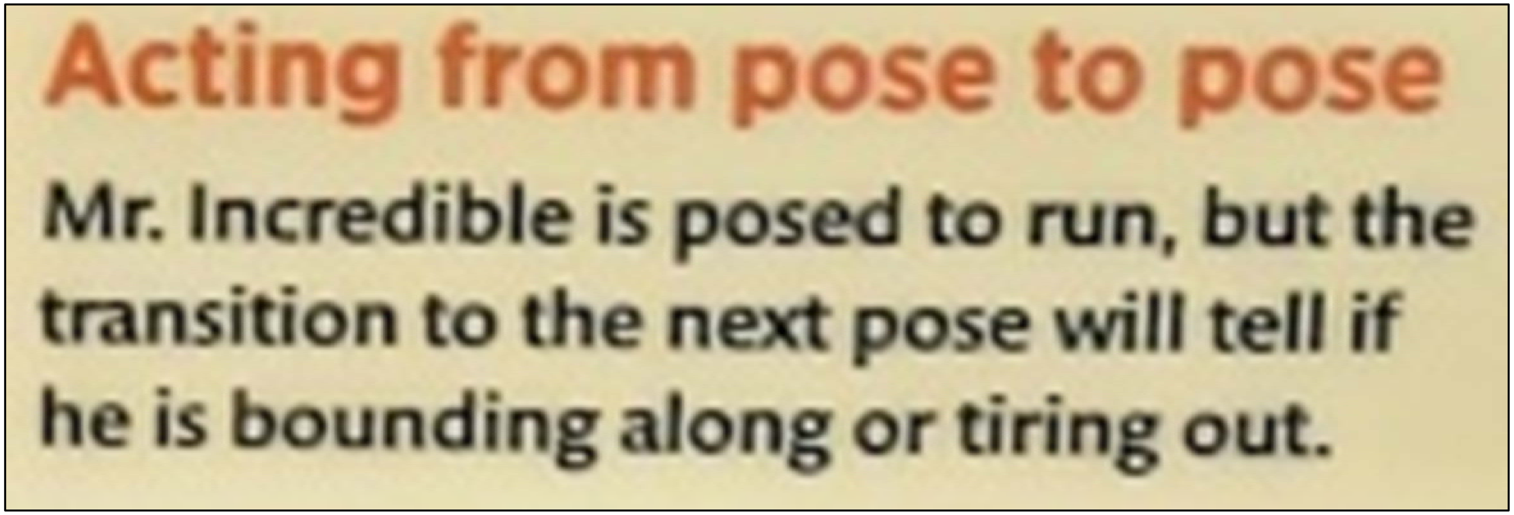
Acting from pose to pose

Mr. Incredible is posed to run, but the transition to the next pose will tell if he is bounding along or tiring out.



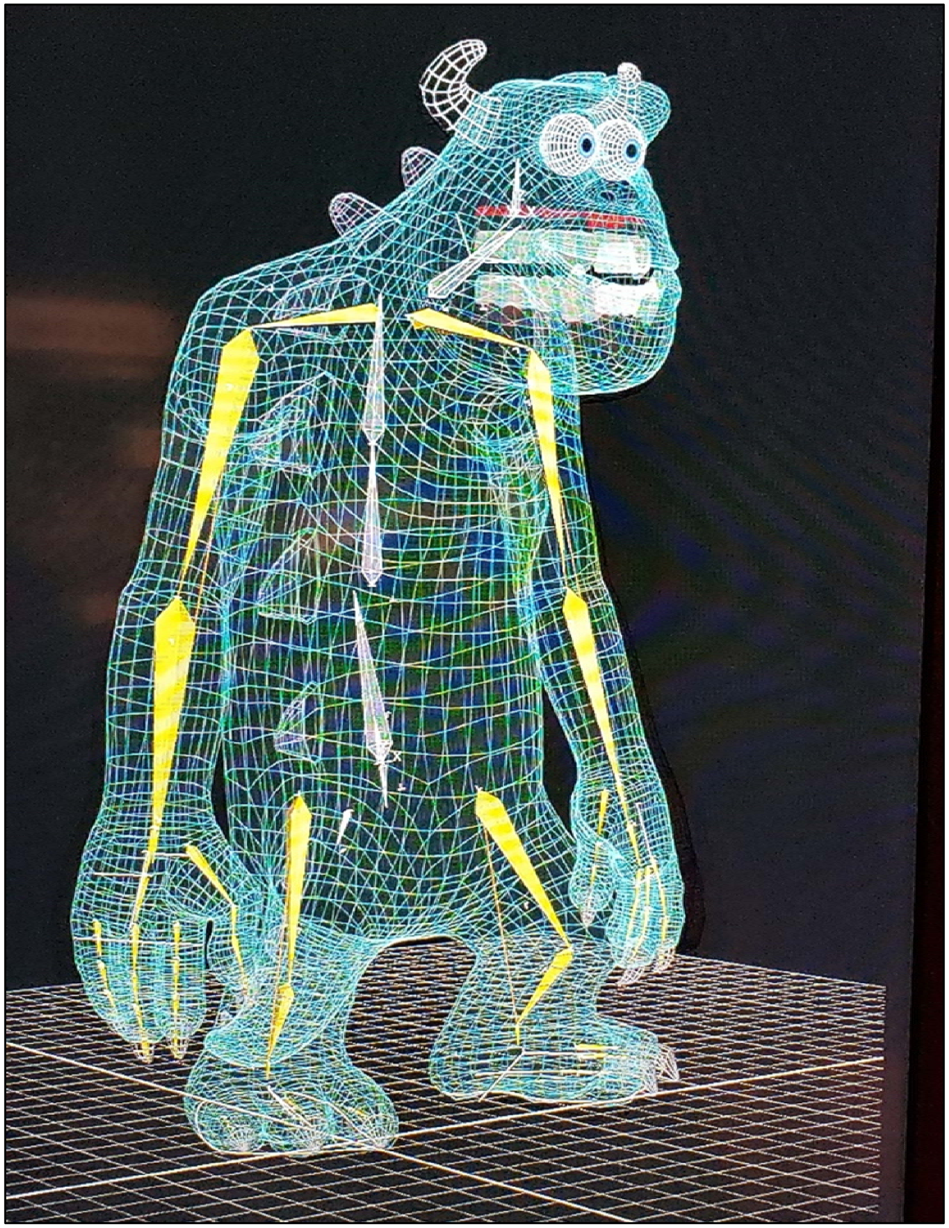
Moving with math

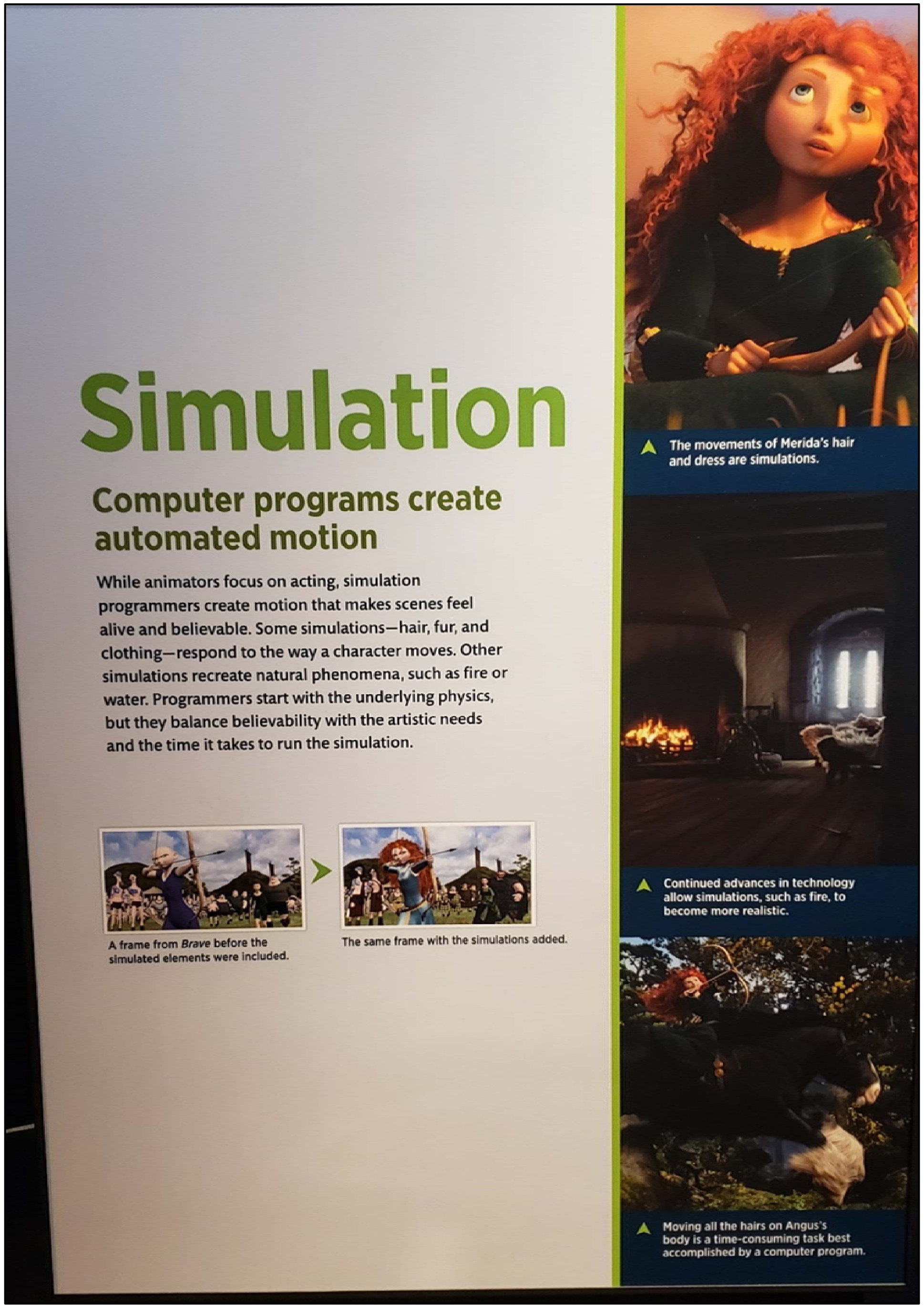
Computer animators position digital models into key poses. Then the computer fills in the transitions based on mathematical functions called splines.



Acting from pose to pose

Mr. Incredible is posed to run, but the transition to the next pose will tell if he is bounding along or tiring out.





Simulation

Computer programs create automated motion

While animators focus on acting, simulation programmers create motion that makes scenes feel alive and believable. Some simulations—hair, fur, and clothing—respond to the way a character moves. Other simulations recreate natural phenomena, such as fire or water. Programmers start with the underlying physics, but they balance believability with the artistic needs and the time it takes to run the simulation.

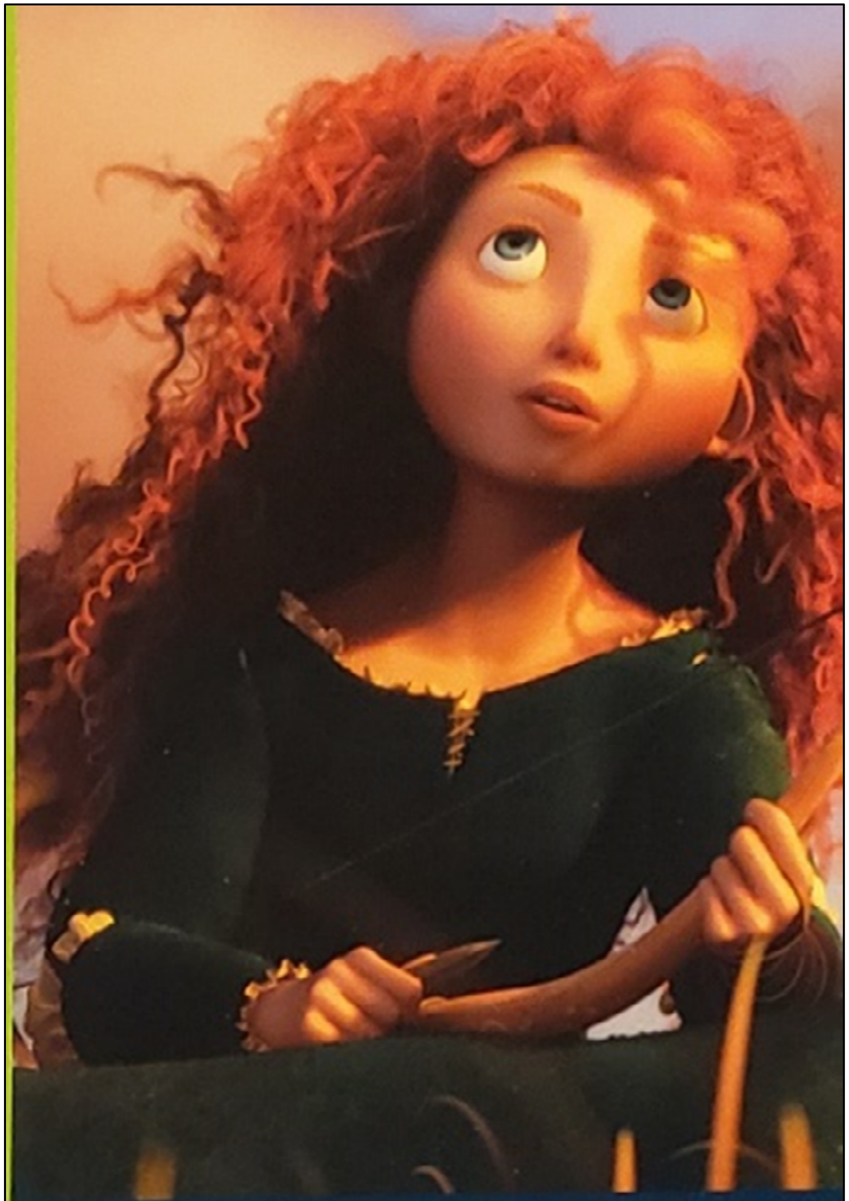
A frame from *Brave* before the simulated elements were included.

The same frame with the simulations added.

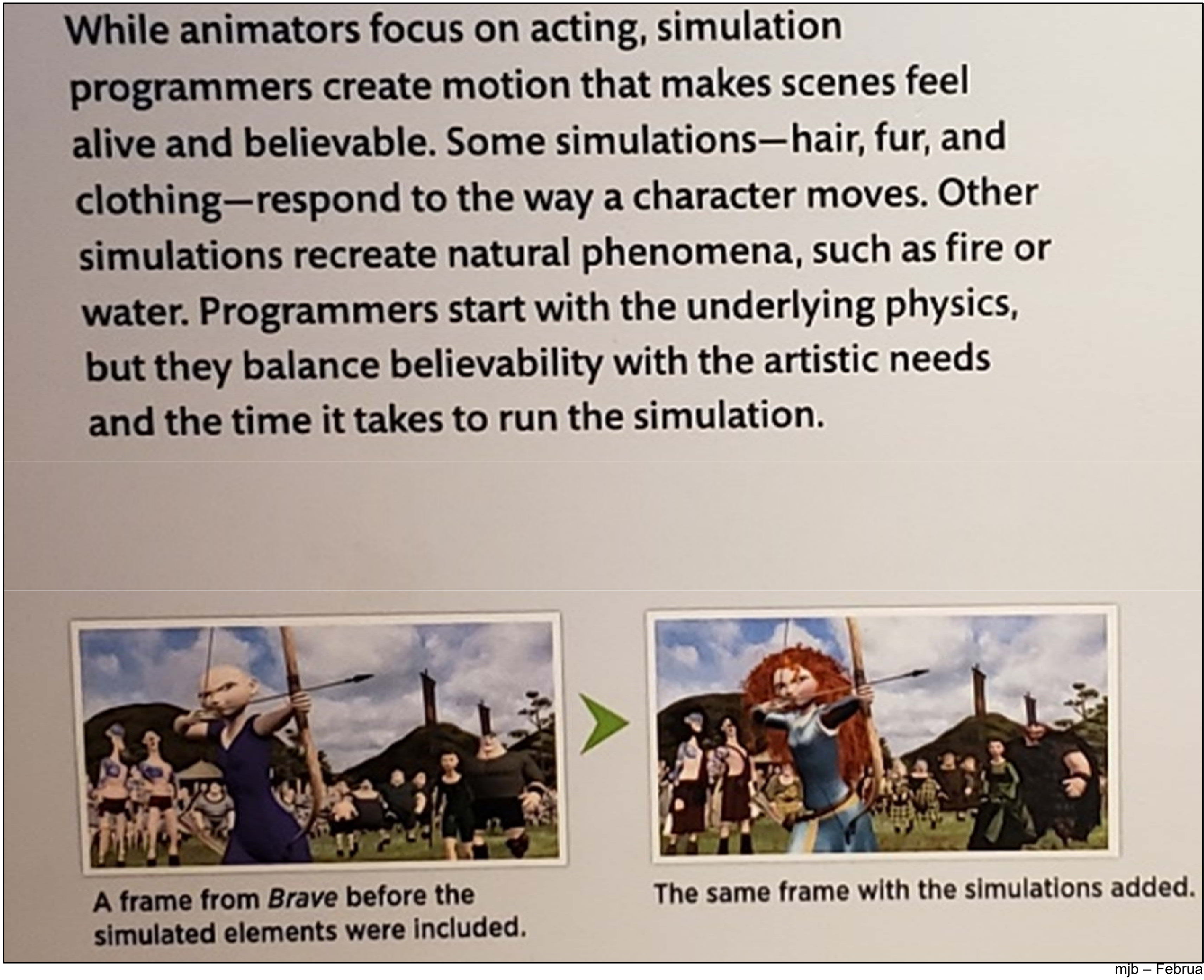
▲ The movements of Merida's hair and dress are simulations.

▲ Continued advances in technology allow simulations, such as fire, to become more realistic.

▲ Moving all the hairs on Angus's body is a time-consuming task best accomplished by a computer program.



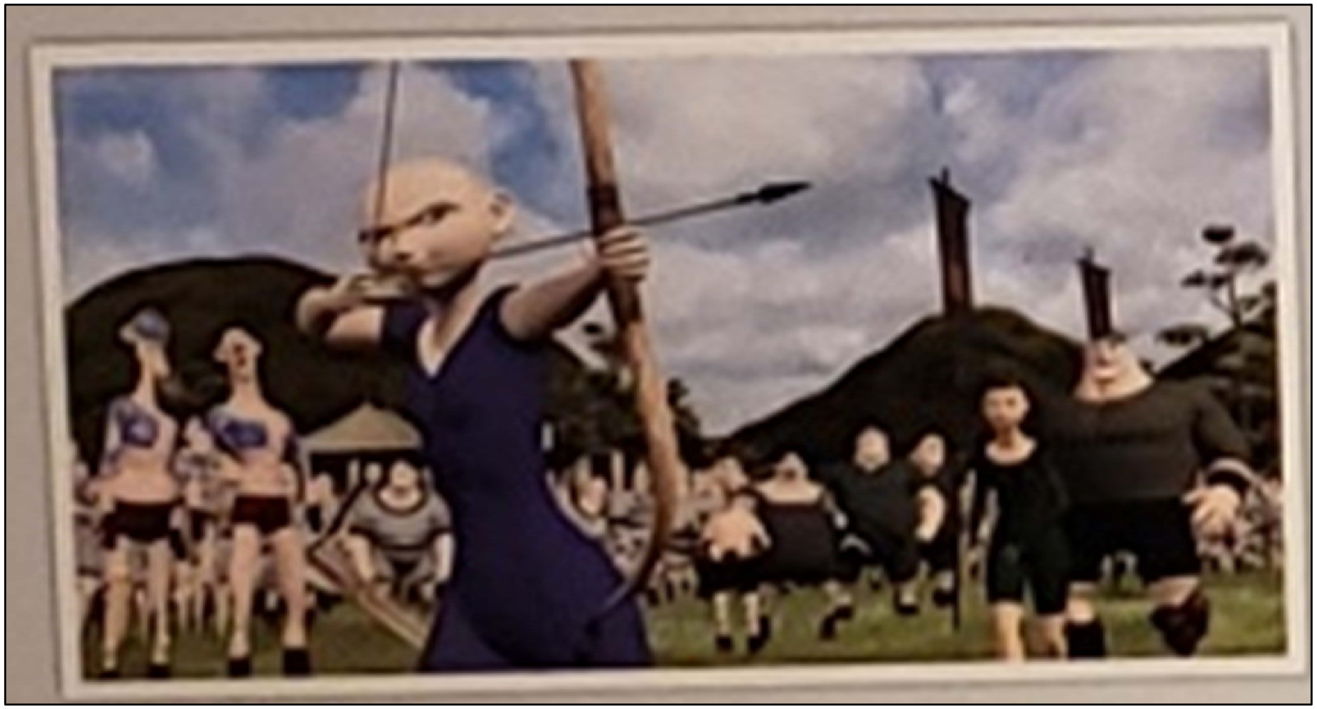
The movements of Merida's hair and dress are simulations.

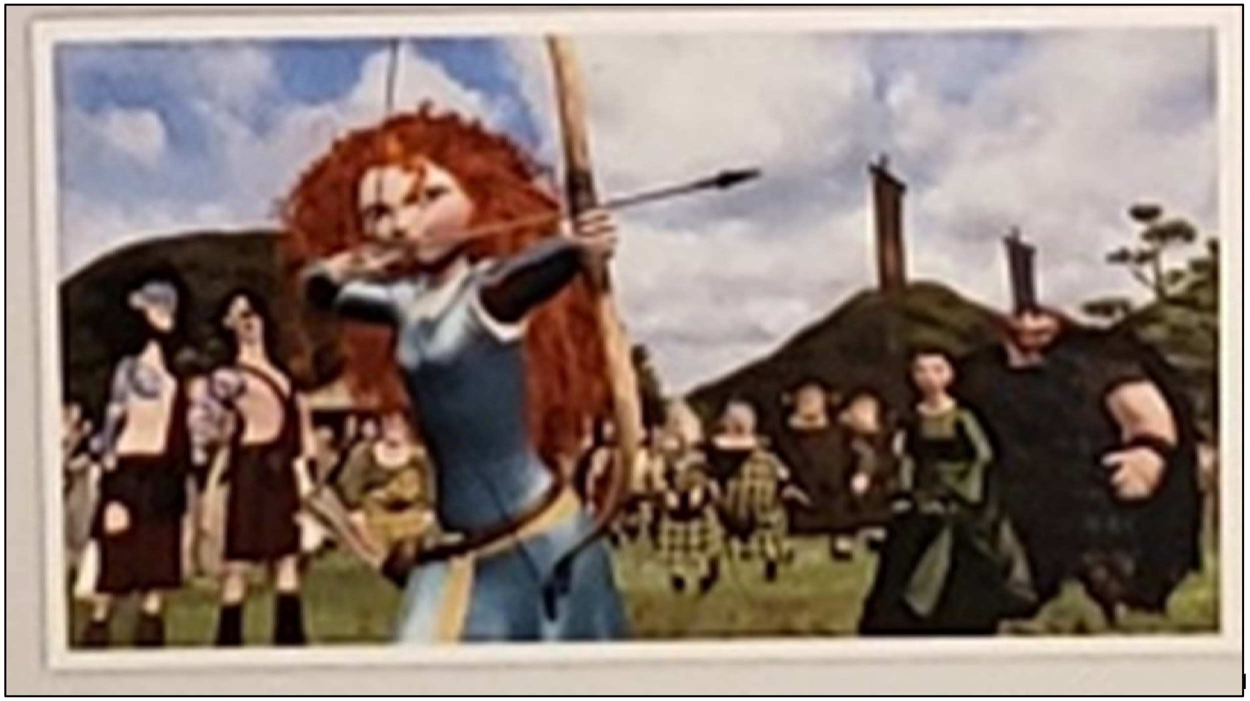


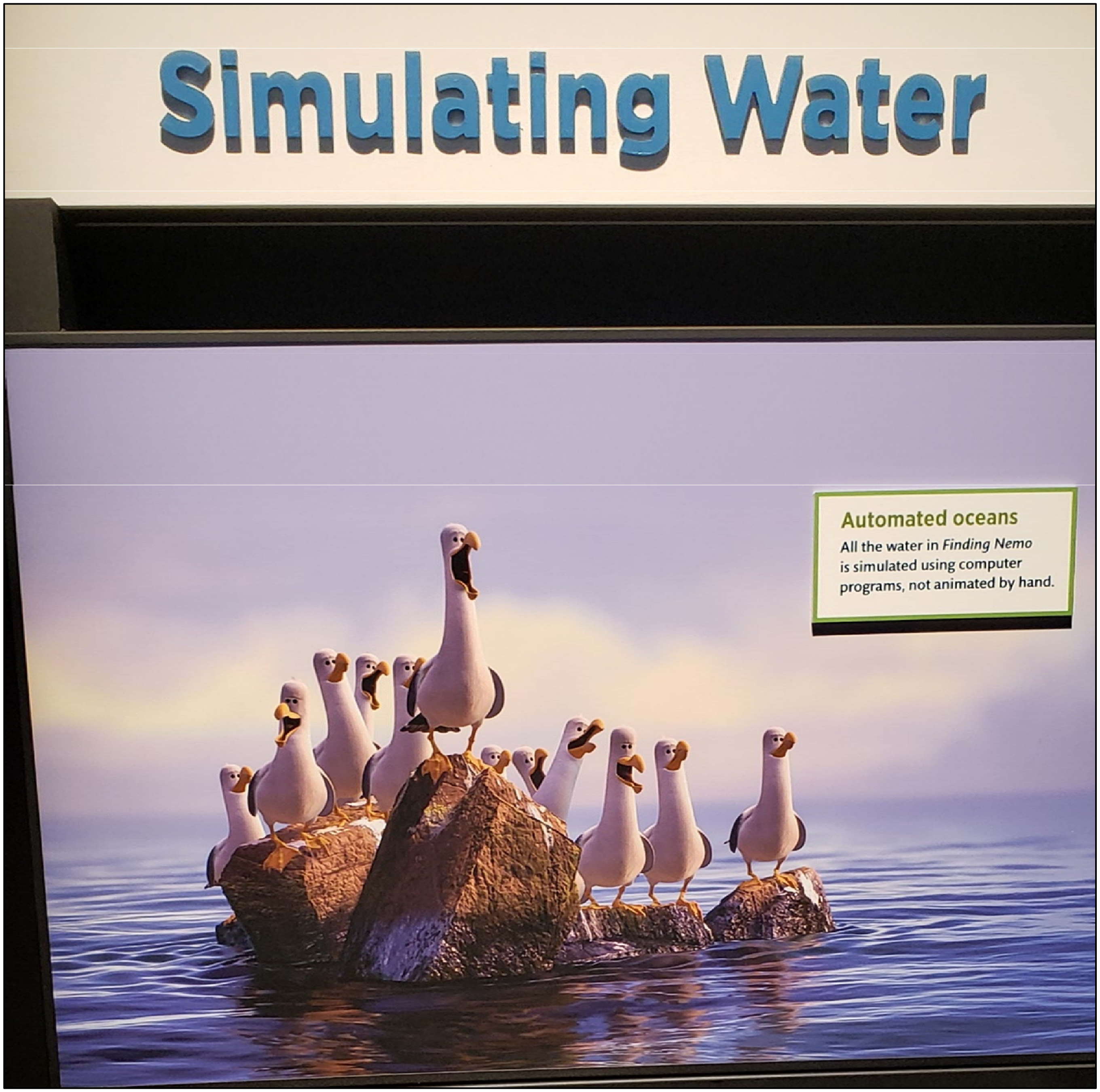
While animators focus on acting, simulation programmers create motion that makes scenes feel alive and believable. Some simulations—hair, fur, and clothing—respond to the way a character moves. Other simulations recreate natural phenomena, such as fire or water. Programmers start with the underlying physics, but they balance believability with the artistic needs and the time it takes to run the simulation.

A frame from *Brave* before the simulated elements were included.

The same frame with the simulations added.



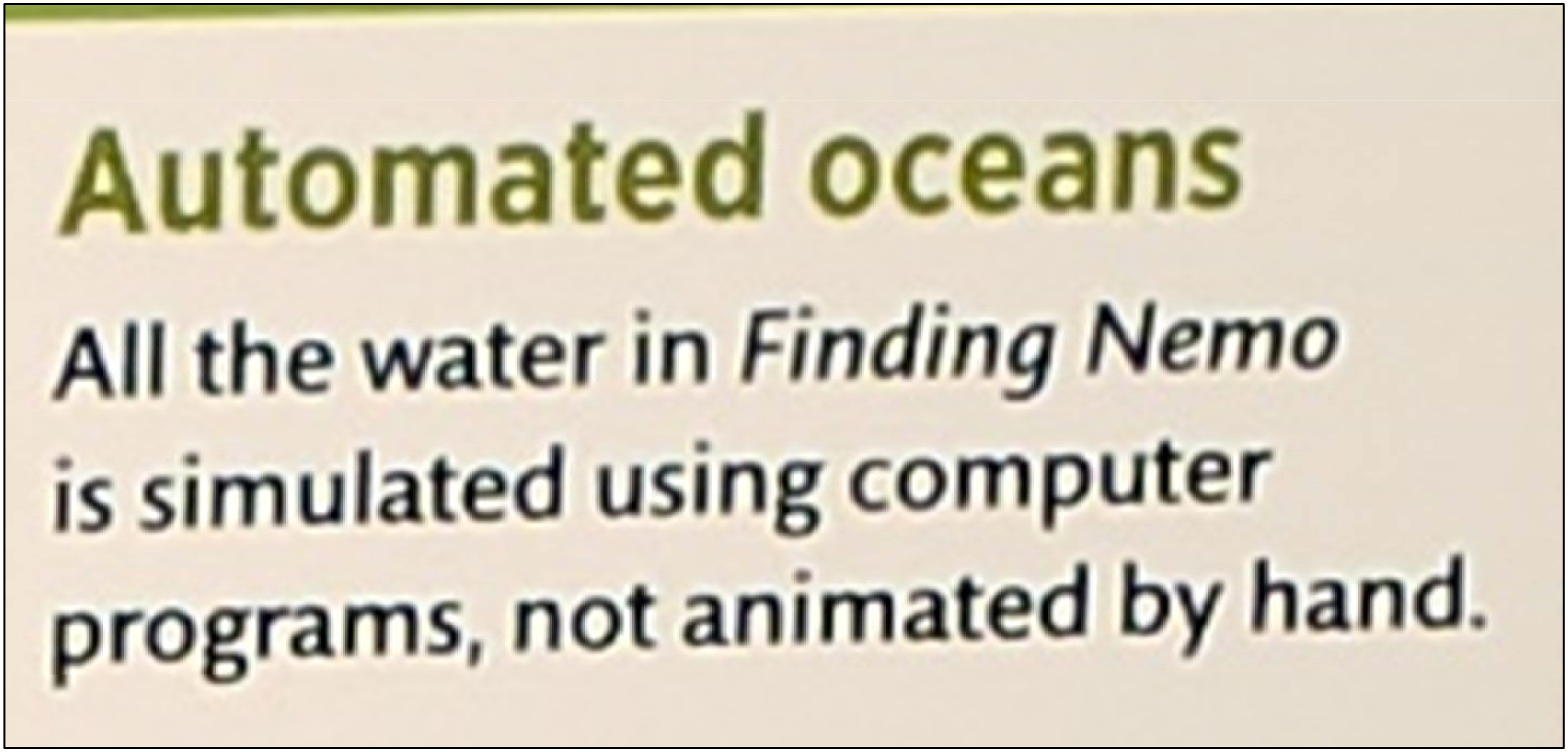




Simulating Water

Automated oceans

All the water in *Finding Nemo* is simulated using computer programs, not animated by hand.



Automated oceans

All the water in *Finding Nemo* is simulated using computer programs, not animated by hand.

The Science of Pixar

At the Oregon Museum of Science and Industry (OMSI)