mantle

an interactive installation

Benjamin D. Richards

regional supremacy assertion ONLINE
STEALTH CAPACITORS
HELM GUIDANCE

What is Mantle?

Mantle is an interactive installation piece designed for medium spaces 3-10m across. It uses a standard monitor and a webcam to track participant movement. When a participant enters its sensory field, a majestic space battle erupts across the screen.

In this way, Mantle makes the participants complicit in the events; they could not happen without interaction. As they discover the way the system works, the audience must decide whether they will enable further destruction, or stop and let a peaceful state develop.

Mantle was born out of my own conflicting impulses. In times of stress, it is tempting to cut loose and unleash furious anger. But is this fury really productive? It feels fulfilling at first, but perhaps it is only making things worse. Mantle encourages user participation through dramatic pyrotechnics, but it is up to the user to decide whether their actions are justified.

Because the piece is intended to provoke contemplation, it is deliberately created without sound. Audio affects an audience on an instinctive level, and a soundtrack of explosions and throbbing space engines would make the piece feel exciting. By removing this component, I encourage the audience to take a more intellectual view of proceedings.

The title of the piece refers to multiple aspects. The Mantle is the cloaking device by which the user's starship avoids detection. The Mantle is the body of the deep sea octopus, whose hunting habits influenced the aesthetics of the space battle. But most importantly, it is the mantle of responsibility which inexorably descends upon the audience, and the choice between conflict or peace.

“Is this fury really productive?”

User Experience

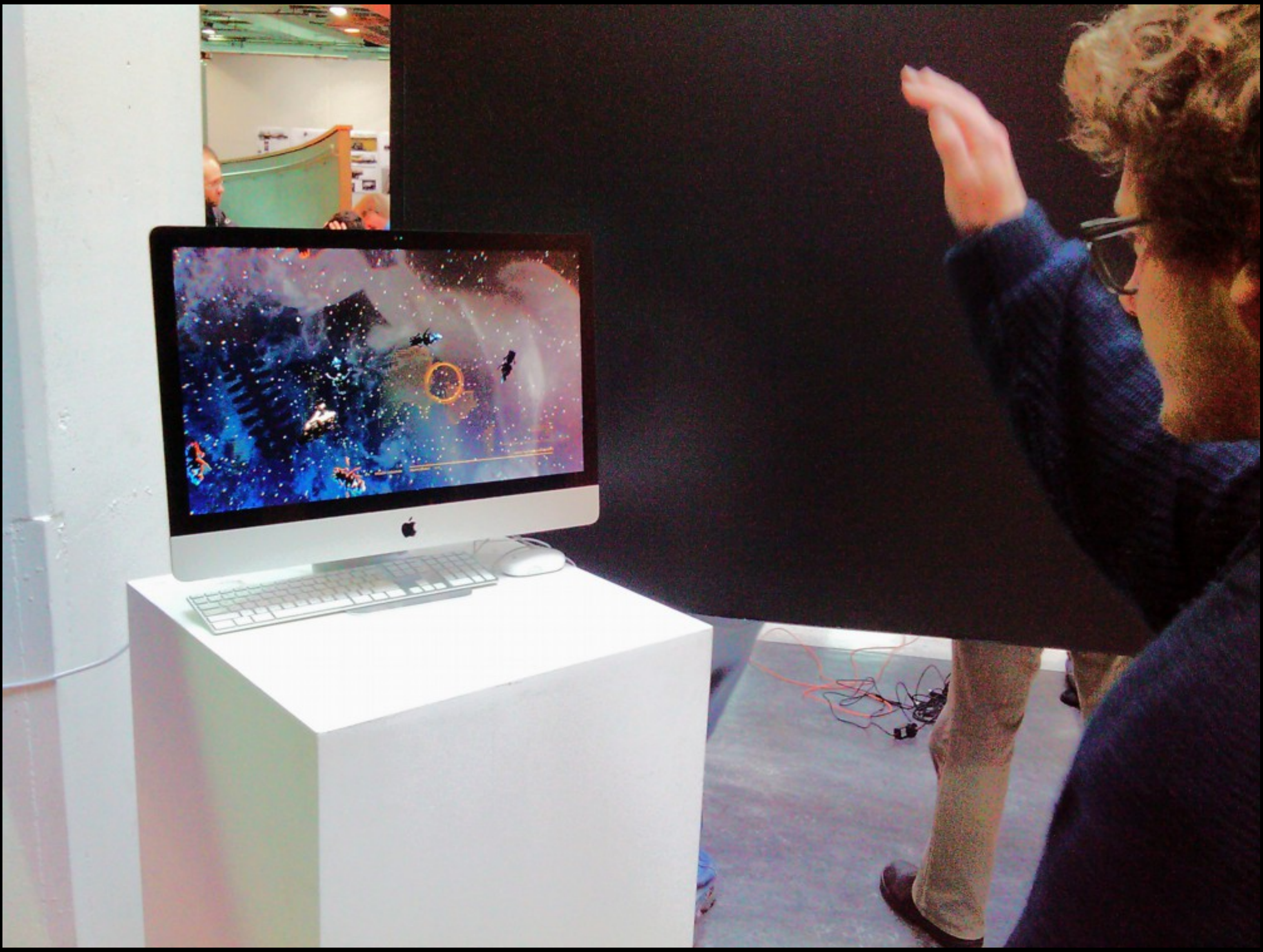
From a distance, the Mantle installation appears in a passive state.

The passive installation presents the user with a view of deep space. Stars glow and dark ships cross the void on some unknown mission. There is little interface visible, and no user ship can be seen.

But upon approaching the installation, the Mantle comes to life. Movement directs a cursor across the screen, and a powerful starship emerges from invisibility to follow it. This ship guns down anything that gets in its way in a series of staggering explosions. A never-ending stream of reinforcements arrive, but they are powerless to resist its weapons.

With some experimentation, the ship can be directed with hand gestures.

When the Mantle reaches the cursor, it is only a matter of time before it enters a power saving state once more, and vanishes from sight, leaving the other ships unmolested. When this happens, the user can either redirect the cursor via movement, or allow the scene to remain at peace. But this realisation likely comes when the user is deep in the visual field. Only by moving very slowly can they escape without triggering another rampage.



Setup

The Mantle installation consists of four vital elements: the program, the display, the lighting, and the camera.

The program must run on a computer with up-to-date Java support. Because of its graphical fidelity it requires a computer with a reasonably powerful graphics card and CPU.

Run the program. It will automatically detect the screen resolution and fill the screen. If performance is choppy, try quitting, reducing the screen resolution and relaunching the program.

The display may be any form of computer output: a monitor, connected TV, or projector. The installation runs in 16:9 aspect ratio so HD resolutions such as 1280*720 or 1920*1080 are recommended. If the display is not in 16:9 the program will run letterboxed, with black bars to the top or sides of the screen. Because the installation features high fidelity graphics it is important to pick the right display for the environment. In a dark room or at night a projector may be effective. In a well-lit space, the higher contrast of a screen is better.

The lighting must separate participants from the background. A well-lit foyer is perfectly sufficient; no special lighting setup is required. In smaller rooms, care must be taken to light the interaction zone correctly. Light should come from the sides or from the direction of the display. This will create contrast between the participants and the background. Care must also be taken to choose a background that contrasts well with human skin hues. This is particularly important in small rooms where light may come from a single bulb. One bulb cannot produce much light and creates dark shadows; while the human eye is comfortable in such conditions the camera will struggle and may need help.

The camera can be any variety of standard webcam. It should be placed facing out from the display. In the case of a monitor it is sufficient to place it on top of the screen. Adjust it so it can clearly see the space before the display. Built-in webcams work fine so long as you adjust the screen. See Operation, below, for information on diagnosing the view directly.

Quick Setup Guide

Ensure your webcam is plugged in. Run the program. See what happens.

Operation

The installation should run without modification. However, it does have controls, listed below.

Press 'S' to take a screenshot. It will appear in a folder called “screenshots” within the program folder. Screenshots are taken in JPG format and named with the date and time.
Press 'ESC' to quit the program instantly. This may not work in the case of unexpected crashes. No such crashes are known at the time of writing, but I suggest that you are confident in force-quitting programs before use, just in case.
Press 'M' to toggle between camera and mouse mode (see below).
Press '~' (or `) to toggle camera buffer diagnostic mode on or off (see below).
Camera/Mouse Mode: The installation is designed to operate with the webcam, but in some cases it may be necessary to use other pointing devices, such as when using the program outside an installation environment, or if the installation is expected to be used by participants with mobility issues. In mouse mode the program uses a normal system pointer, usually a mouse.

If the program does not detect a camera upon launch, it will lock itself into mouse mode. If a camera is subsequently connected, relaunch the program to detect it.

The current interface mode is indicated by an icon on the lower right.

Camera Buffer Diagnostic Mode: The installation uses a webcam to run a motion detector, which may require calibration. In diagnostic mode, the various buffers of the camera are displayed along the left side of the screen. These buffers can be tuned with the number keys: press the indicated number to increase the value, and press the number with 'SHIFT' to decrease.

This program was written for English keyboards; the function may differ on other language layouts. The explicit values are given below for reference.

1 / !: Tumble input blur. The camera feed is blurred to eliminate vibrations and minor movements. Change this to alter the blur amount.

2 / @: Tumble learn rate. The program quickly learns its surroundings so it can pick out movement. Change this value to alter the learning rate. By default it is quite high to cope with quick motions. By reducing it to a value of 0.01 or lower, it may instead start to recognise objects instead of movements.

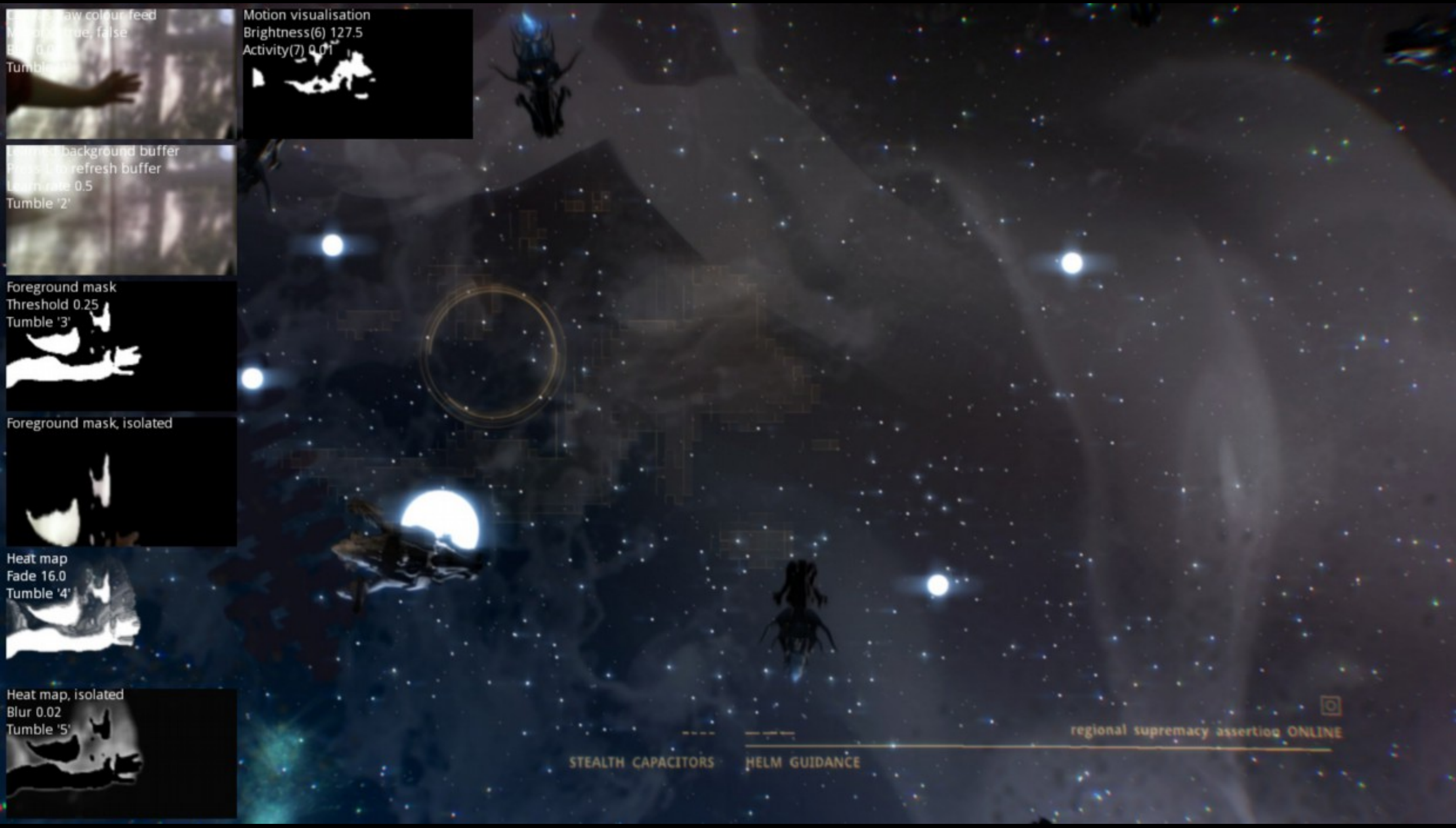
3 / #: Tumble foreground mask threshold. This is the measure of difference that tells the program whether something is not part of the background. Lower values make the program more sensitive, but are also prone to flicker and interference.

4 / $: Tumble heat map fade. This is a system that remembers recent motion in an area, smoothing out the user's motions. Lower values make the program respond faster, but may reduce sensitivity.

5 / %: Tumble heat isolation map blur. This subtracts the foreground mask from the heat map, leaving only motion trails around the edges. Thus only movement is detected, and still objects make no difference. Lower values reduce the size of the movement trace, making it harder to sense motion.

6 / ^: Tumble motion cursor brightness threshold. This measures the sensitivity of the cursor to motion. Lower values make it more sensitive, but prone to flicker.

7 / &: Tumble motion cursor activity threshold. This is the minimum ratio of active pixels to total picture before the cursor will move. Lower values make the cursor more sensitive, but prone to flicker.



Computer Vision

The Mantle installation uses a camera to perceive the world around it. I haven't taught the computer to think for itself; it's just a series of images overlaid and buffered over time. The camera feed is recorded as a “learned background”, and differences with the current frame can be used to determine what's moving and what's new.

This computation is mostly performed on the graphics card, greatly accelerating the drawing process.

The installation displays an abstraction of the machine vision overlaid across the screen as part of the HUD (Heads Up Display). This is an important part of user feedback. By seeing their own motions reflected, they gain an understanding of the nature of the interface. However, the overlay is not so bright as to overwhelm the action, and fades away automatically when the user is still.

Deep Space

The action of Mantle takes place in the void between the stars. Starships clash against a background of ancient wars.

The background is designed to give contrast with the foreground action. Deep darkness and brilliant light interact with the optical effects of the starships. The background was created using The GIMP, a freeware image editing program, and visual assets available from the Image*After website.

The starships were designed to evoke creatures of the deep sea, yet retain a sense of mechanical purpose. They are intricate machines composed of many parts.

When conflict comes, those parts explode in space-rending detonations. An explosion may only last a fraction of a second, but that fraction is the key to the entire experience. It must be dazzling enough to temp the user to repeat it, but convincing enough to terrify and instil doubts. I spent a long time tuning explosion physics, filling them with detail and colour to draw the eye.

Starships are differentiated through colour and light. The user should have no difficulty identifying “their” ship. This is achieved purely through coloured lighting, as all ships have the same material qualities.

Creating the Starships

The starships in Mantle were created in Autodesk Maya, then exported as a series of texture maps for use on sprites.

This geometry is taken into the Transfer Maps tool, then applied to a simple square. The result is a texture that holds all the detail of the 3D model, but is much faster to render.

Each piece of the starship is transferred separately. There are 14 pieces in a single ship, from tiny manoeuvring thrusters to soaring motor-arms that sway with the ship's passage.

Each component in turn has up to five layers. These are the diffuse, specular, normal, emissive, and warp layers.

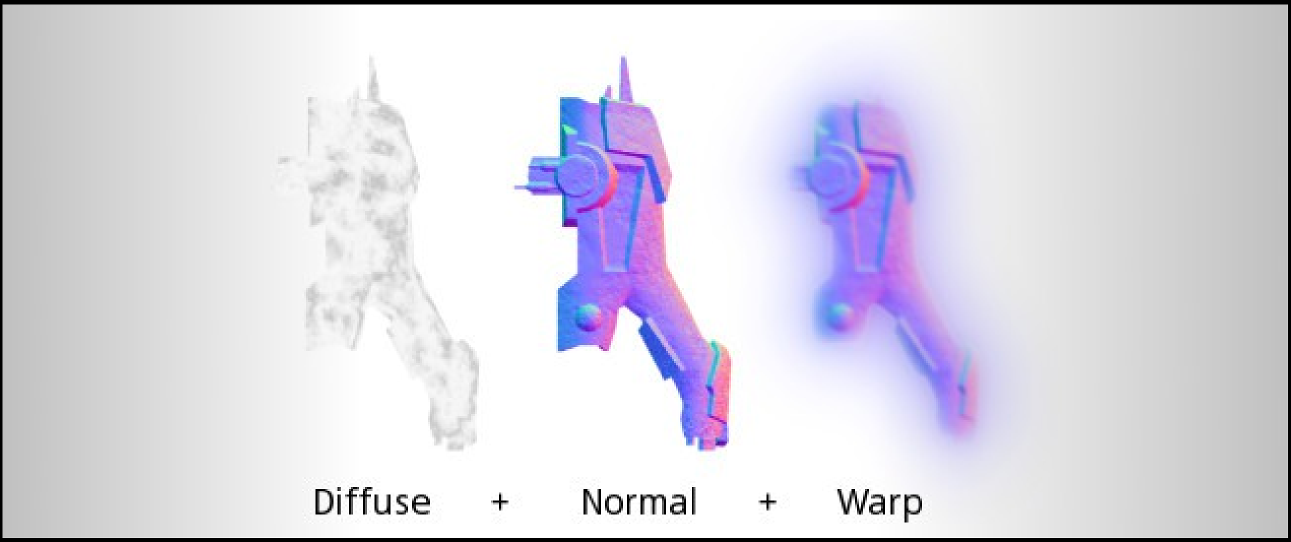
The diffuse layer contains simple colour information. It has no lighting or shadows. This is the “base” layer.

The specular layer contains information about shininess and reflectivity. High specular areas reflect the light with brilliant highlights.

The normal layer contains information about the surface topography: how it curves and at what angle light bounces off. This is used to light the layer without recourse to complicated 3D geometry.

The emissive layer contains information about objects that glow under their own light. These elements are similar to diffuse elements, but are unaffected by lighting.

The warp layer is similar to a normal layer, but uses its information to distort the underlying image instead. The vectors stored in the image push the image around, causing the illusion of distortion.

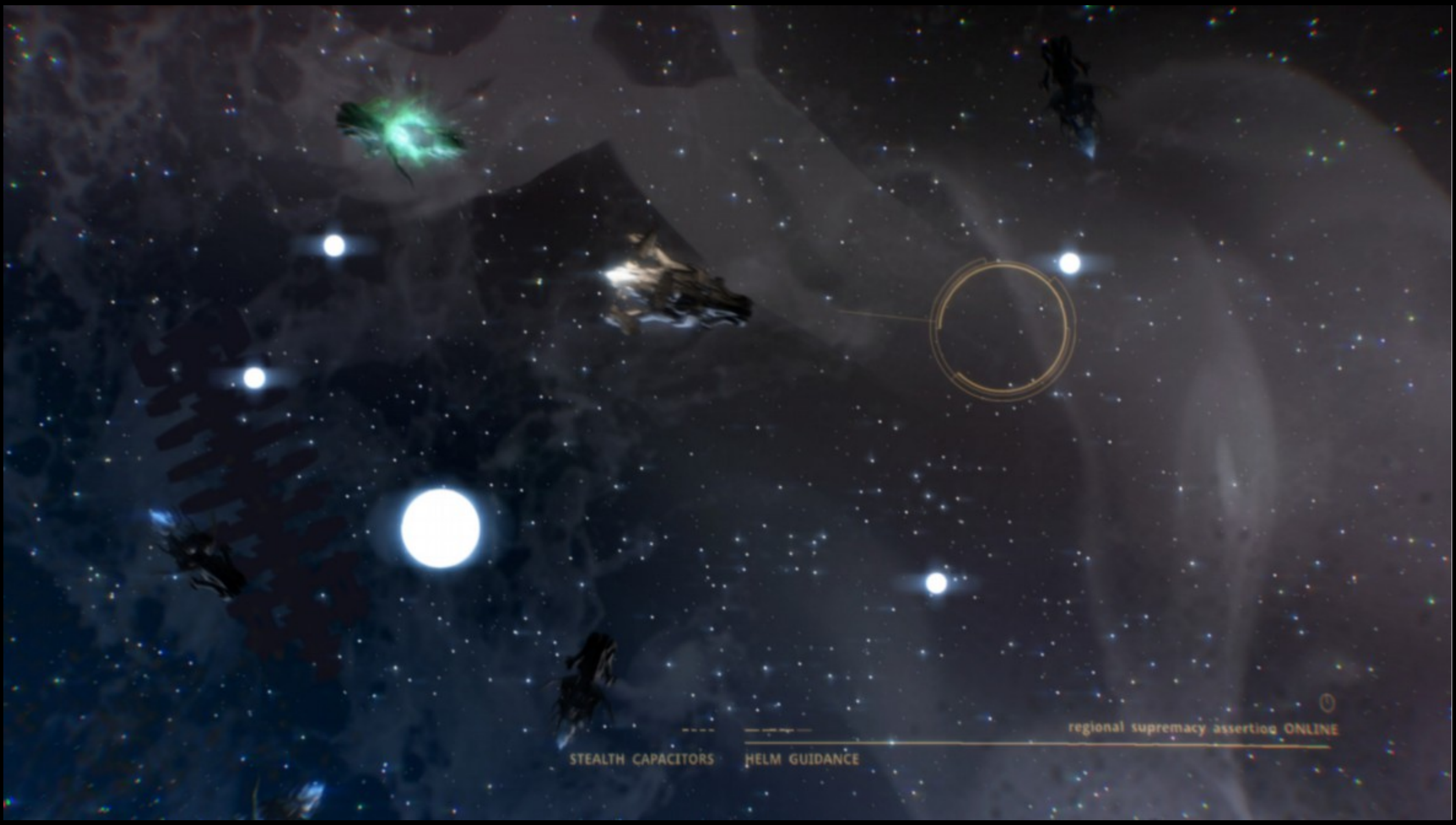


These layers are fed into a render management algorithm in the program. There, they are drawn into a series of buffers in a process called deferred lighting. By storing the information, I can apply lights instantly and in great number, as simply as drawing stencils onto the screen.

Simplifying with Shaders

The layers of a starship component are fed into a render management algorithm in the program. There, they are drawn into a series of buffers in a process called deferred lighting. By storing the information, I can apply lights instantly and in great number, as simply as drawing stencils onto the screen.

Finally, the information is composited together with a little magic, and makes its way to the screen. A starship meets its demise in dramatic fashion and the cycle continues.



About the Creator

My name is Benjamin D. Richards. I have a background in 3D animation and am currently completing a Masters qualification in Media Design Innovation at Victoria University of Wellington.

For questions about this installation or other topics, check me out at:

benjamindrichards@gmail.com

www.benjamindrichards.com

And check out the source code for Mantle on GitHub at:

https://github.com/BenjaminDRichards/MDDN_442_Project2