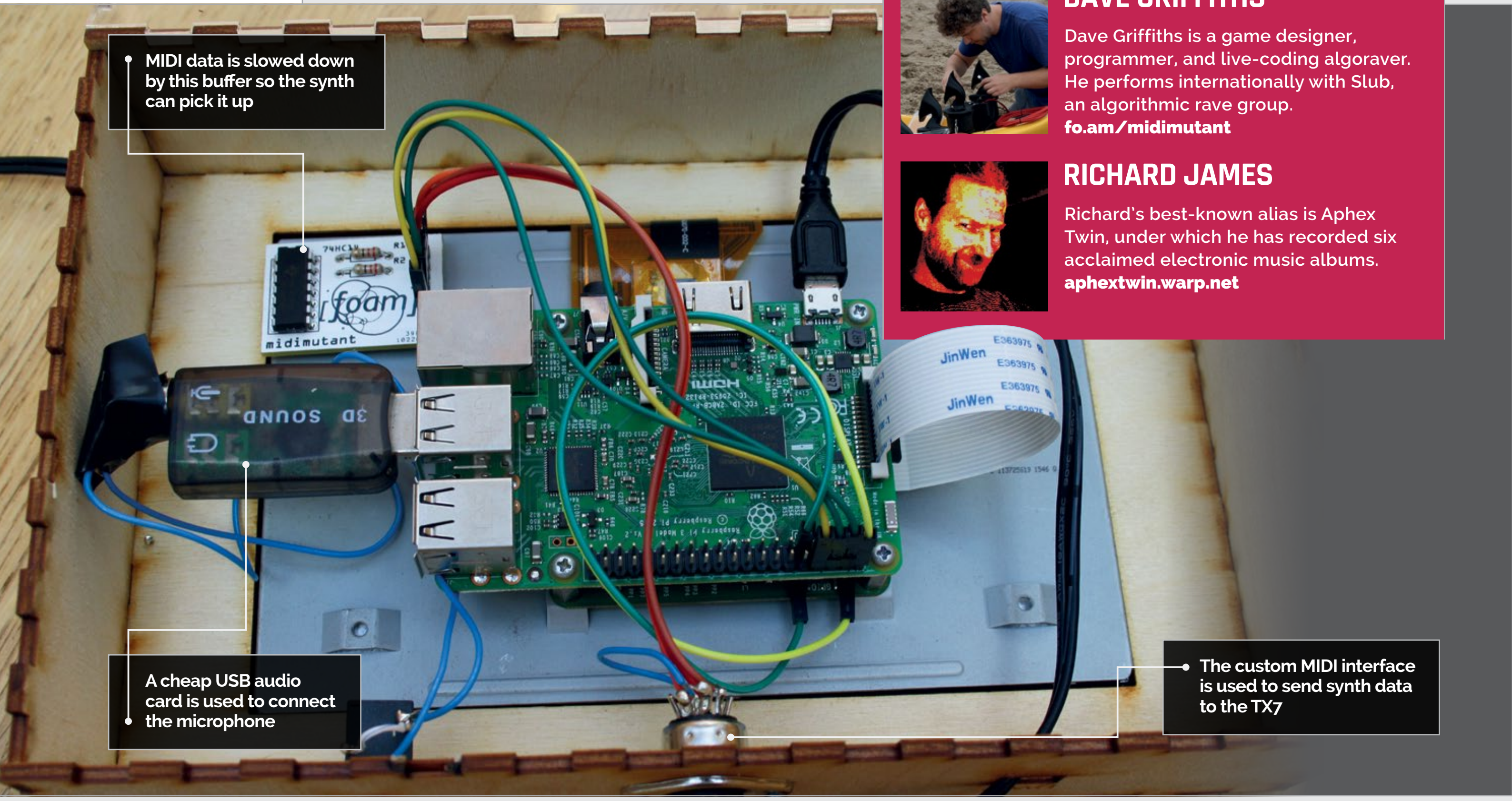


MIDI data is slowed down by this buffer so the synth can pick it up

A cheap USB audio card is used to connect the microphone

The custom MIDI interface is used to send synth data to the TX7

DAVE GRIFFITHS

Dave Griffiths is a game designer, programmer, and live-coding algoraver. He performs internationally with Slub, an algorithmic rave group.
fo.am/midimutant

RICHARD JAMES

Richard's best-known alias is Aphex Twin, under which he has recorded six acclaimed electronic music albums.
aphextwin.warp.net

APHEX TWIN MIDIMUTANT

Somewhere in Cornwall, there's a synthesizer that boinks endless strange sounds as they slowly evolve towards a new synth patch. **Sean McManus** investigates

Quick Facts

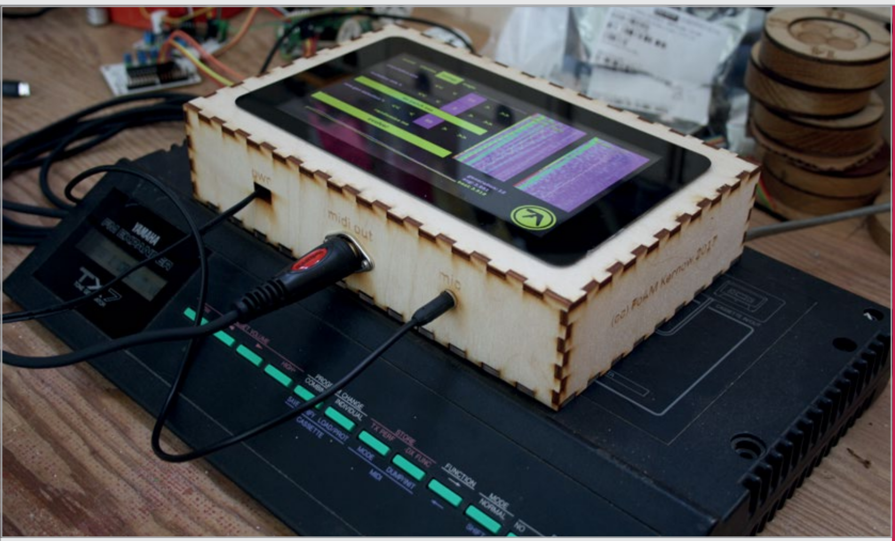
- Making a good patch manually can take days
- Midimutant can make thousands of patches per day…
- …But only about half of them are useful
- Dave coded Midimutant in C++
- He now has 3792 saved Midimutant DX7 patches

If you're a fan of '80s pop, you'll have heard the Yamaha DX7. It's one of the best-selling synthesizers, and its built-in sounds can transport you to the era of big hair and neon leggings. One reason it is evocative is that so many artists used those preset sounds, to the point of them becoming a cliché. The synth is based on frequency modulation (FM), and it could take a few days to create a good patch (or sound). Many chart-toppers took the easy way out.

"I'm nuts about FM synthesis," says Richard James, who creates electronic music as Aphex Twin. "The first proper synth I got was a DX100 and I've always thought there's got to be a more interesting way to program the damn things than laboriously going through all the hundreds of parameters. Even though I quite like doing that anyhow, hehe."

A conversation with his friend Dave Griffiths led to Dave building Midimutant, a Raspberry Pi-based device that programs the synth for you. "This is something you want to leave running for hours," says Dave. "The Raspberry Pi is essential because you can make a stable setup that isn't going to start updating itself and reboot."

The idea came from a lost feature on the Kyma synth, based on the work of Andrew Horner, which enables sounds to evolve. Midimutant uses a similar approach: you give it a sound, and it aims to recreate it on the Yamaha TX7, a version of the DX7 without a keyboard. "There's nothing especially great about a TX7 apart from I love all forms of FM and that is a really small unit, so it's a lot easier to handle than



Above Midimutant sits on top of the Yamaha TX7 synth. The Raspberry Pi Touch Display is used for control

a DX7 or other FM devices I've got lying about," says Richard.

Dave adds: "The DX7 is the 'classic' FM synth. If it works on that, it would work on anything."

Random scoring

At the start, a population of random sounds is created. Each one is compared to the source sound using Mel-frequency cepstral coefficients (MFCC), a way of comparing sounds that comes from recent work on speech recognition.

> The DX7 is the 'classic' FM synth. If it works on that, it would work on anything

Each of the random sounds is scored and ranked for its similarity to the source. The program gets rid of the lowest scoring half, and creates a new population by copying with errors (mutation) and crossbreeding (mixing together) the top half. "Repeat this process tens or hundreds of times and it usually (although not always) converges on something good," says Dave. The most interesting sounds emerge from trying to match beats and voices.

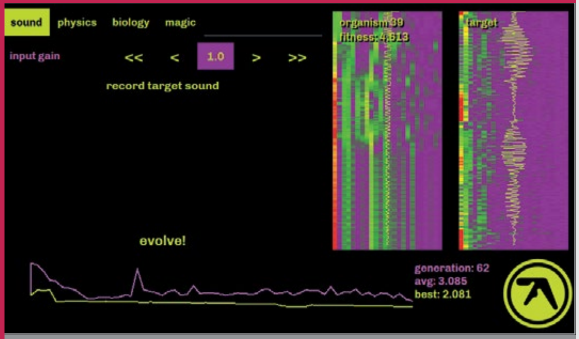
One fascinating aspect of the project is that Midimutant doesn't need to understand how the synth functions. It just needs to know how to format the (initially random) data so the synth can use it, and it uses the sound the synth produces to score the results.

The biggest challenge? "Listening to endless bonkers sounds for days on end, changing parameters slightly and waiting ages to see if there was an improvement," says Dave. "Actually, that was quite fun..."

The Yamaha DX7 is an iconic synth, but Midimutant should work with other FM synths too

Image credit: Flickr user deepsonic (CC BY 2.0)

MAKING MIDI SOUNDS WITH MUTATION



input gain << < 1.0 > >>

organism 39
fitness: 4.613

generation: 62
avg: 3.085
best: 2.081

>STEP-01

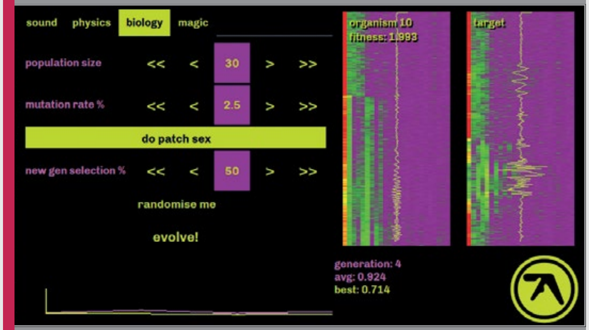
Record your sound

Use the USB microphone to record the source sound you want Midimutant to try to match. Short sounds will evolve faster.

>STEP-02

Run the algorithm

Midimutant creates a random set of synth patches. It sends the data to the synth using MIDI. The synth plays the sound, and Midimutant 'listens' with the microphone.



population size << < 30 > >>
mutation rate % << < 2.5 > >>
new gen selection % << < 50 > >>

organism 10
fitness: 1.993

generation: 4
avg: 0.924
best: 0.714

>STEP-03

Do patch sex!

The sounds are compared with your original. The best ones are evolved with random mutations. Use the 'patch sex' option to combine sounds, too. For best results, leave it running overnight.