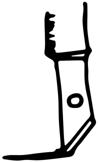
Making machines that make

Nadya Peek, infosyncratic.nl

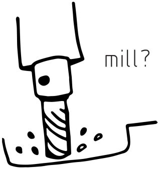
30c3, Hamburg, December 2013

So you want to make a machine!

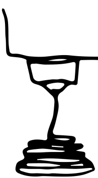
What does it do?



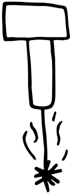
cut?



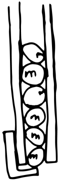
mill?



extrude?

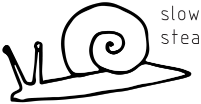


burn?

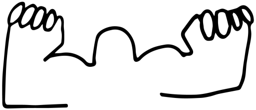


dispense
candy?

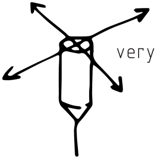
Depending on what you want to move, you may need different methods...



slow and steady?

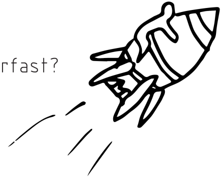


lots of force?

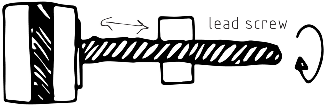


very precise?

superfast?

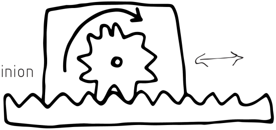


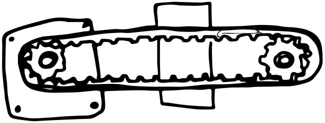
If you want to move something, you need to select an appropriate drive train, like:



lead screw

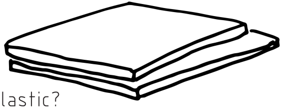
rack and pinion



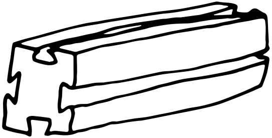


timing belt

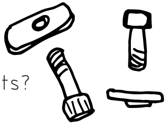
A frame holds your machine together, and you could make one out of lots of different kinds of materials!



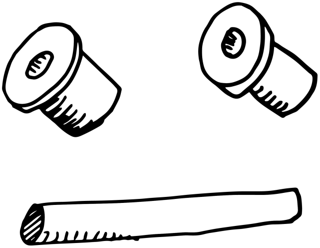
routed plastic?
laser cut wood?



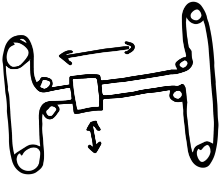
aluminum extrusion?



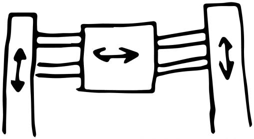
t-slot nuts?



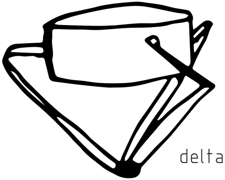
The drive train's motion needs to be restricted in the axes you want to move in. You can do this with guide shafts, tracks, cable guides, linkages, and many other ways.



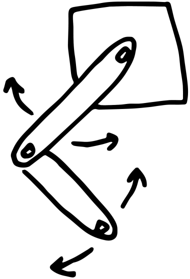
h-bot



serial kinematics

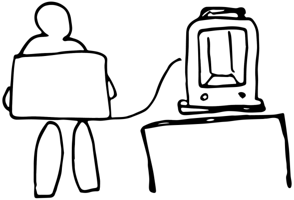


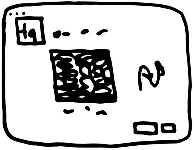
delta bot



polar coordinates

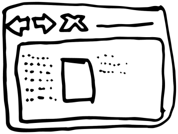
Finally, how are you going to control the machine? There are different kinds of software to stream machine code to machines with, how do you want yours to work?



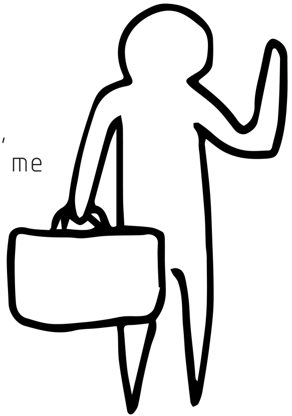


Maybe a drawing program interfaces directly to the machine

Maybe the machine is controlled from a browser...



I made my machine,
it is just right for me

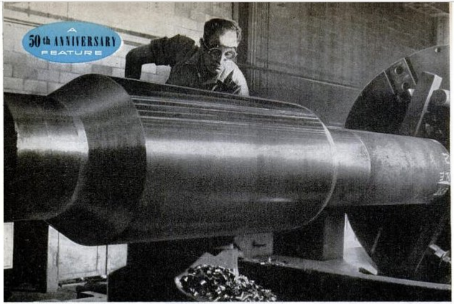


Pyrmont Bridge

Automobile Showroom, Montreal Canada

Chullora Aircraft Factory circa 1943

(B.586)



U. S. Steel Corp. photo

By George Scullin

Giant steel roll soon to take its place in a U. S. Steel rolling mill is machined on a 60-inch lathe in the forging shop at the firm's Homestead District works

The jet engine and the wrist watch, the power saw and the 1952 automobile— all are products of those modern wonders—

THE TOOLS THAT MAKE TOOLS

AROUND early March this year, a few newspapers announced casually that the Air Force had been given the green light on the purchase of 20 machine tools. It was just a small story and the editors couldn't get too excited about it. Not even at the size of the machines, four stories high; or their cost, $389,000,000! Stories like that are routine in this year of 1952.

But what if that story, through some slip in the time machine, had appeared before the young editors preparing the first issues of *Popular Mechanics* in 1902? What about it would be strange?

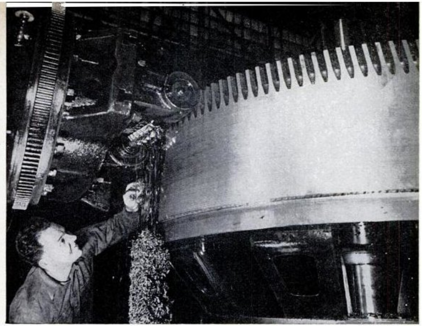
Not the words "Air Force," though man had yet to fly a power-driven aircraft. These farseeing young men were already convinced that man *would* fly, and soon, and that there would be an Air Force. Not the size of the machines. These editors were dedicating their magazine to the conviction that the years to come would produce mechanical wonders beyond anything even dreamed of at the turn of the century. To anticipate these marvels and explain them in words and pictures their readers could understand was the job they had already created for themselves. But we do think they would have been stunned by the $389,000,000.

In 1902 that sum would have bought the year's output of the entire machine-tool industry, would have bought the industry as well, and there would have been enough left over to put a little white fence around the whole thing. In fact, the industry was so small that few people had ever heard of it, and fewer still knew what it was.

Yet this is the tiny industry that has made possible our entire way of life. Without it, we would be living on the products of our bare hands, with a standard of living approaching that of Colonial days.

What are these machines that produce all this magic? Well, they are a weird family. They are the tools that make the tools that make everything else. But, being a family, they also make each other. This makes them the only self-procreating race in the machine world.

To understand the huge, fantastic, almost-human machine tools of today, let's take a look at their ancestors as the first editors of *Popular Mechanics* knew them.

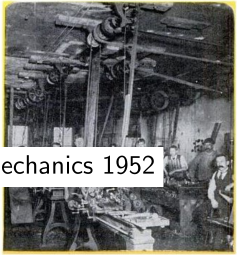


Westinghouse photo

Hobbing machine cuts teeth in a gear over eight feet in diameter; chips pile up beneath the cutting tool

A lot of selective breeding has gone into them since, but basically they remain unchanged. In 1902 a machine tool was described as a nonportable, power-driven tool that shaped metal by removing surplus material in the form of chips.

The first and oldest of these tools was the lathe. By 1902 a more flexible version called the turret lathe was coming into popularity. The next was the drill, which was fine for drilling holes in metal but no[...] enough [...] toleranc[...] 1902. F[...] inch, a boring machine was used. Today drilling and boring are words often used interchangeably, but to the

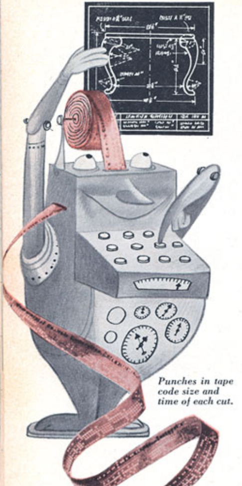


Machine-tool shop of 50 years ago presented maze of overhead pulleys. This was National Acme Company

In an electronic lab at MIT, engineers now are

Teaching Power Tools to Run Themselves

By Hartley E. Howe



Punches in tape code size and time of each cut.

SO JOE WORKSHOPPER figures he'd like to turn out a set of dining-room chairs—and at the same time break in his new Model 100 Super Tapemaster. Joe whips down to the hardware store and looks over photographs of different designs. He settles on a Swedish pattern popular 'way back in 1955—delicate and handsome, but full of difficult reverse curves.

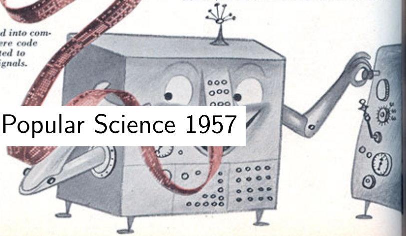
That doesn't worry Joe. He plunks down $10 for a week's rental of a batch of tapes—one each for legs, arms, back and seat.

That night, he clamps a nice piece of birch into his Tapemaster, slips the tape into the control box, flips the switch, and sits back with his pipe and the new issue of *Outdoor Life*.

Forty minutes later, the rumble of the Tapemaster stops and Joe takes a look. One leg is finished. So he clamps on another piece of birch . . .

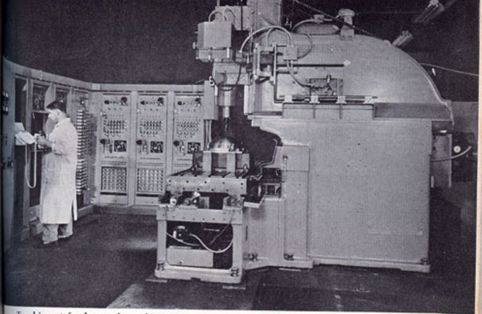
Sure it's a dream—in 1955. But the engineering basis for Joe's Tapemaster exists right now. Sitting up in the Servo-mechanisms Laboratory of the Massachusetts Institute of Technology in Cambridge, Mass., is a milling machine that will turn out any metal part at the command of a little roll of tape. Originally a standard, vertical 28" Cincinnati Hydro-Tel, it now has hitched to it $50,000 worth of electronics.

To conceive, design and build the MIT machine took some quarter-million



Tape is fed into computer where code is converted to electric signals.

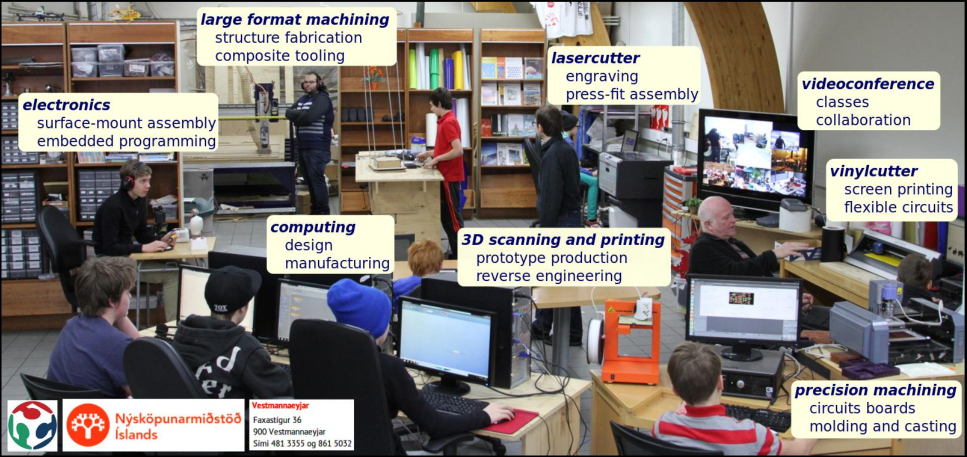
Popular Science 1957



Too big yet for home shop, this MIT milling machine is run by computer-control at left.

Signals control three-dimensional movement of cutter head, time each cut.

Andreas Gursky


large format machining
structure fabrication
composite tooling
lasercutter
engraving
press-fit assembly
videoconference
classes
collaboration
electronics
surface-mount assembly
embedded programming
vinylcutter
screen printing
flexible circuits
computing
design
manufacturing
3D scanning and printing
prototype production
reverse engineering
precision machining
circuits boards
molding and casting
Nýsköpunarmiðstöð Íslands
Vestmannaeyjar
Faxastígur 36
900 Vestmannaeyjar
Sími 481 3355 og 861 5032

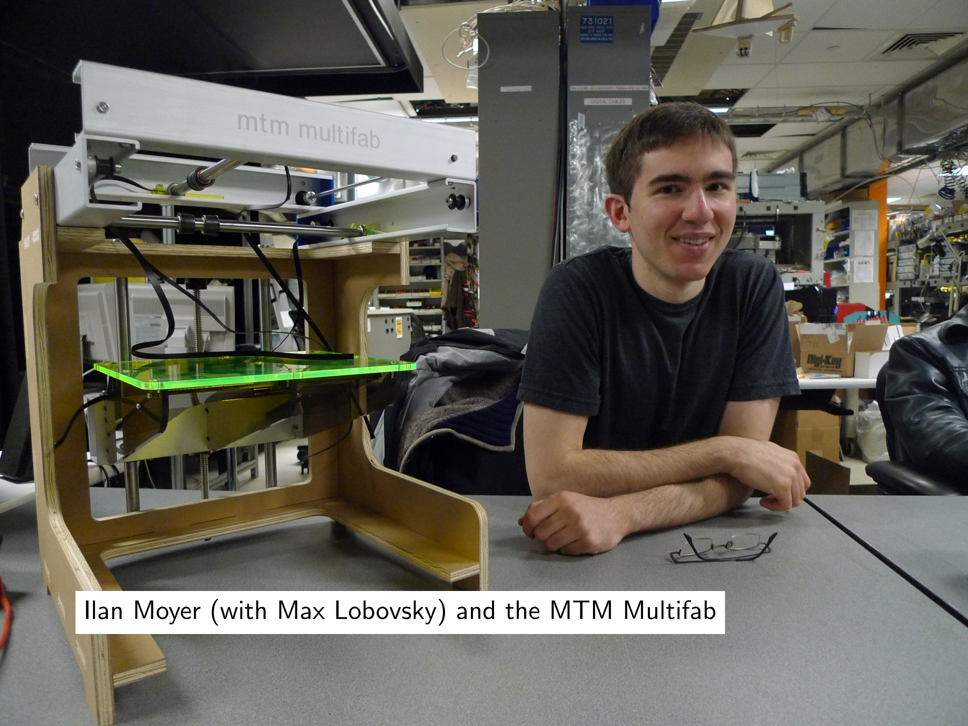
How to make (almost) anything

How to make something

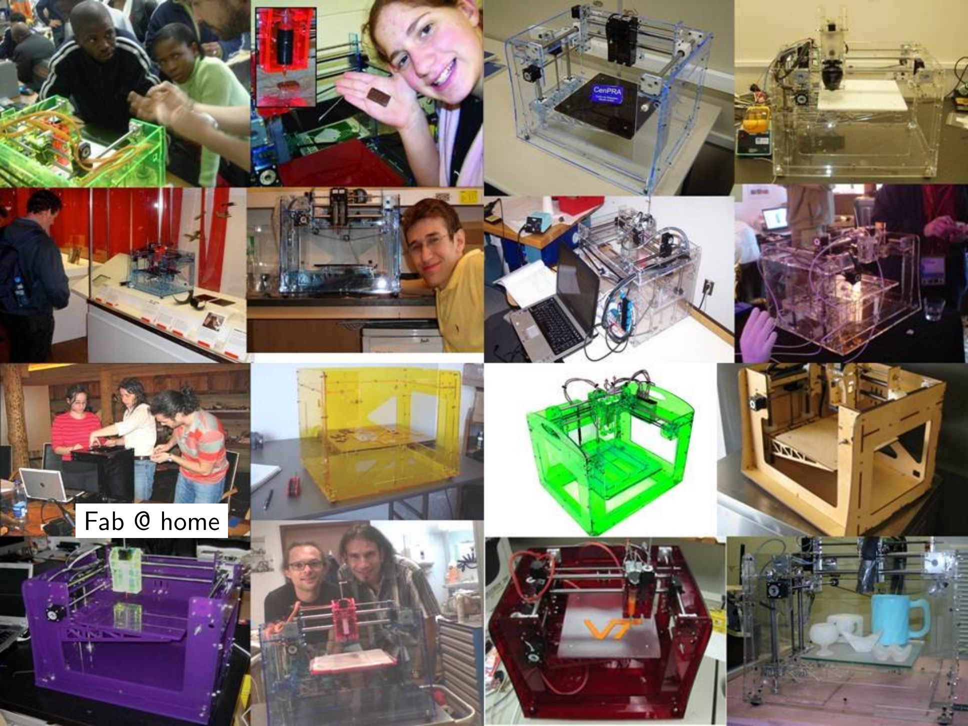
that makes

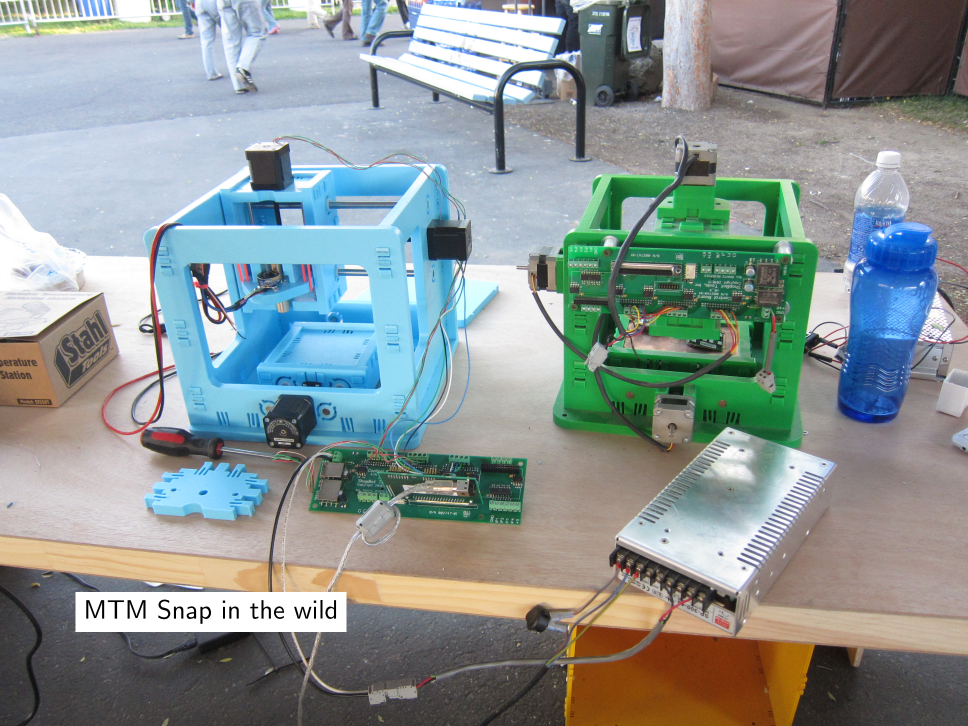
(almost) anything

Jonathan Ward and the MTM A-Z

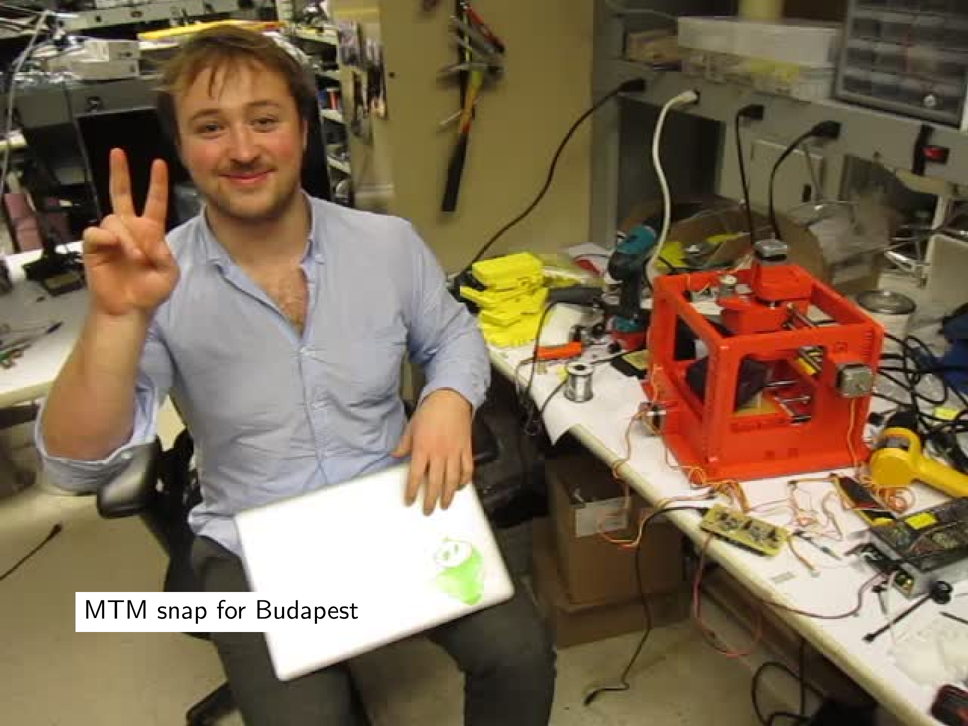


Ilan Moyer (with Max Lobovsky) and the MTM Multifab


Fab @ home
CenPRA

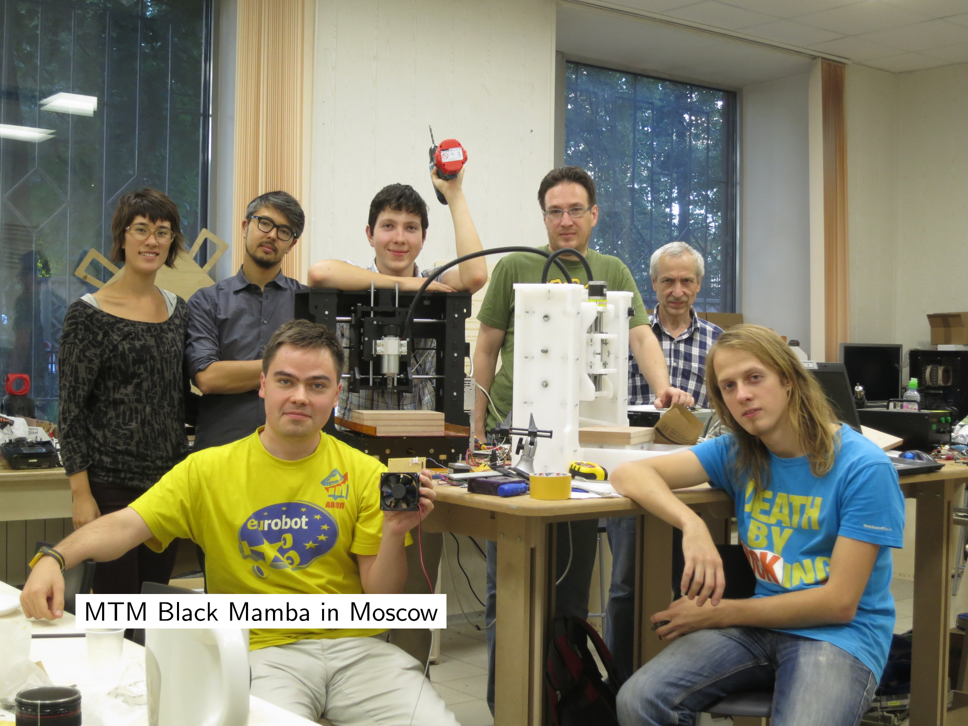


MTM Snap in the wild

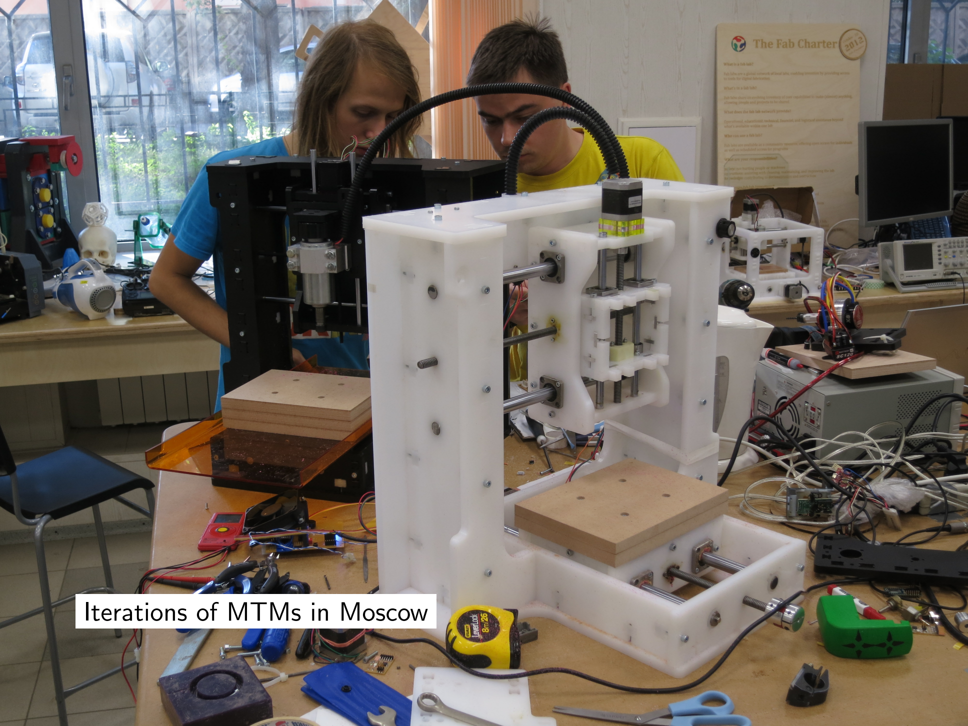


MTM snap for Budapest

MTM Snap for Maharastra

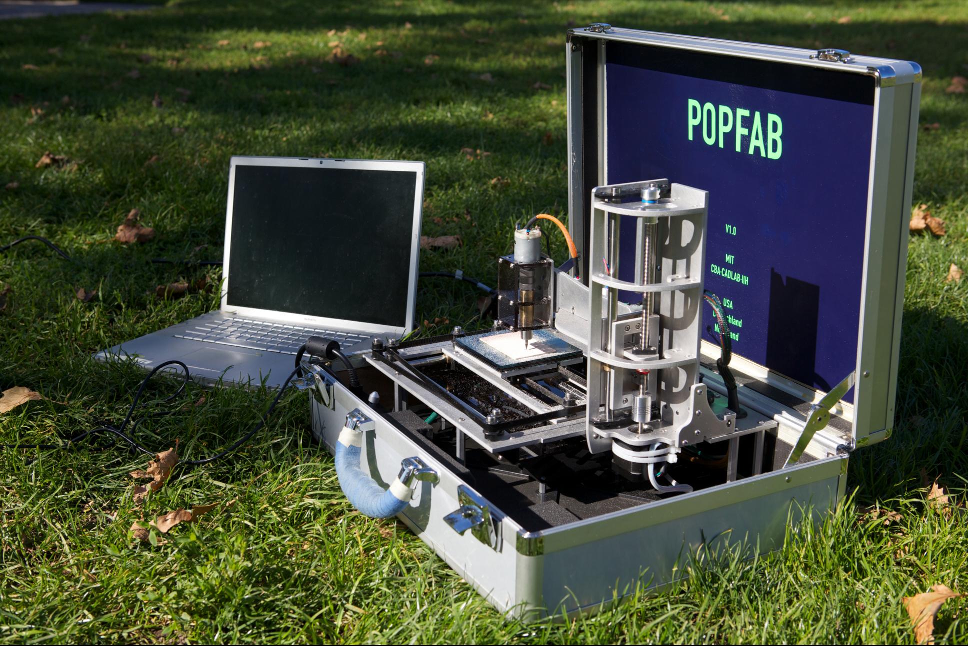


MTM Black Mamba in Moscow

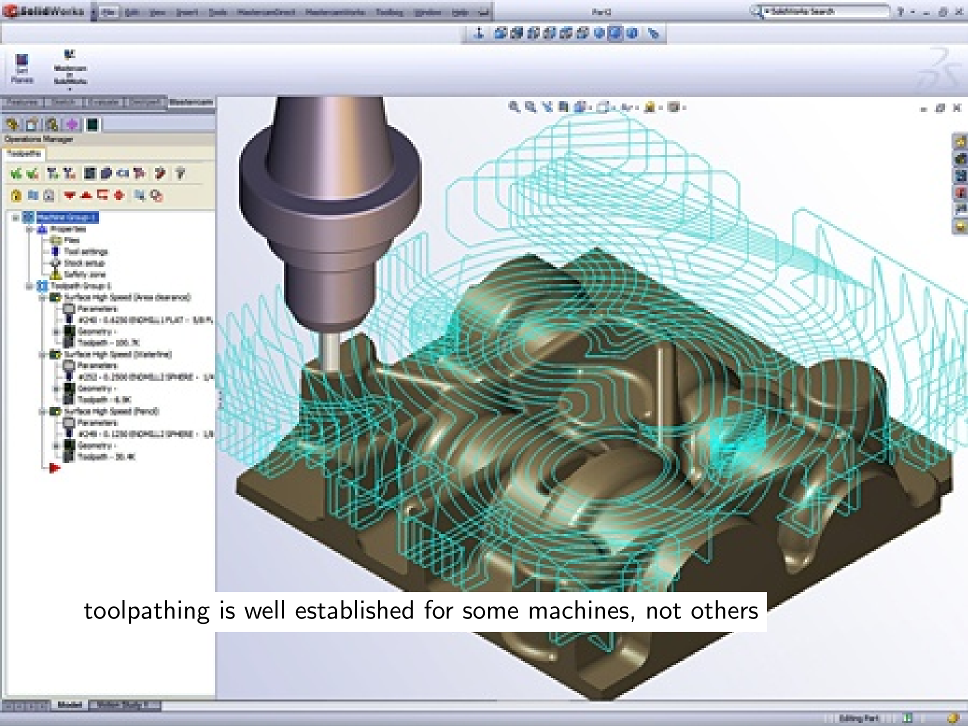


Iterations of MTMs in Moscow

Maybe the form
isn't quite right


POPFAB
V1.0
MIT
CBA-CADLAB-IIH
USA

Maybe something else
isn't quite right



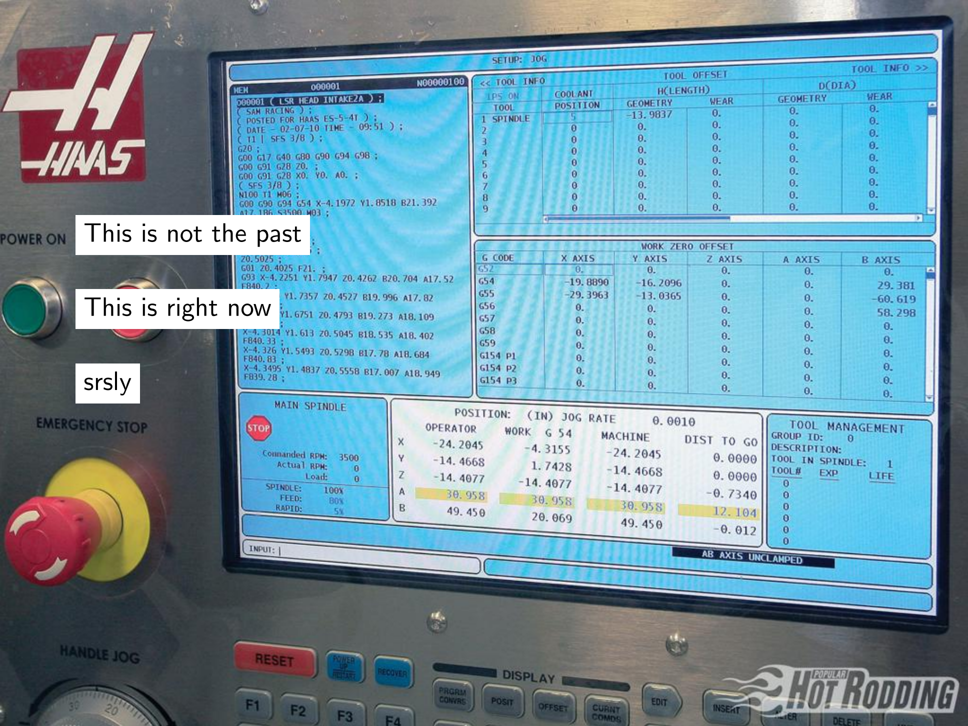
toolpathing is well established for some machines, not others

G-code	Functions	G-code	Functions
G0	Rapid positioning	G53	Move in absolute machine coordinate system
G1	Linear interpolation	G54 à G59	Use fixture offset 1 to 6, G59 to select a general fixture number
G2	Clockwise circular / helical interpolation	G61	Exact Stop mode
G3	Counterclockwise circular / helical interpolation	G64	Constant Velocity mode
G4	Dwell	G73	Canned cycle - drilling - fast pullback
G10	Coordinate system origin setting	G80	Cancel canned cycle mode
G12	Clockwise circular pocket	G81	Canned cycle - drilling
G13	Counterclockwise circular pocket	G82	Canned cycle - drilling with dwell
G15	Polar Coordinate moves in G0 and G1	G83	Canned cycle - peck drilling
G16	Cancel polar Coordinate moves in G0 and G1	G84	Canned cycle - right hand rigid taping (not yet implemented)
G17	XY plane select	G85	Canned cycle - boring, no dwell, feed out
G18	XZ plane select	G86	Canned cycle - boring, spindle stop, rapid out
G19	YZ plane select	G87	Canned cycle - back boring (not yet implemented)
G20	Inch unit	G88	Canned cycle - boring, spindle stop, manual out
G21	Millimeter unit	G89	Canned cycle - boring, dwell, feed out
G28	Return machine home (parameters 5161 to 5166)	G90	Absolute distance mode
G30	Return machine home (parameters 5181 to 5186)	G91	Incremental distance mode
G28.1	Reference axis	G92	Offset coordinates and set parameters
G31	Straight Probe	G92.1	Reset G92 offset and parameter
G40	Cancel cutter radius compensation	G92.2	Reset G92 offset but leave parameters untouched
G41	Start cutter radius compensation left	G92.3	Recall G92 from parameters
G42	Start cutter radius compensation right	G93	Inverse time feed mode
G43	Apply tool lenght offset (plus)	G94	Feed per minute mode
G49	Cancel tool lenght offset	G95	Feed per revolution mode
G50	Reset all scale factors to 1.0	G98	Initial level return after canned cycles
G51	Set axis data input scale factors	G99	R-point level return after canned cycles

g-code is limited and needs to be interpreted


advanced file transport
i.e. a usb stick


end mills
unsurprisingly



This is not the past

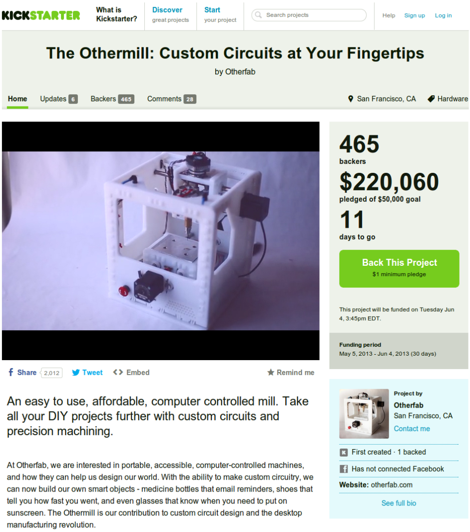
This is right now

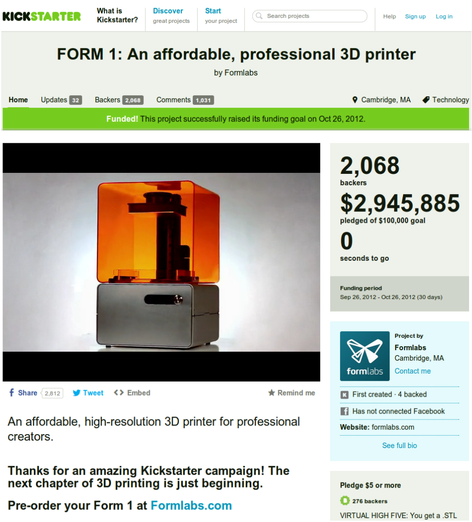
srsly


HURCO
VM10
MAX
BROOKS
NEVER
watching
machines having all the fun

using computers


watching
machine having all the fun
Roland



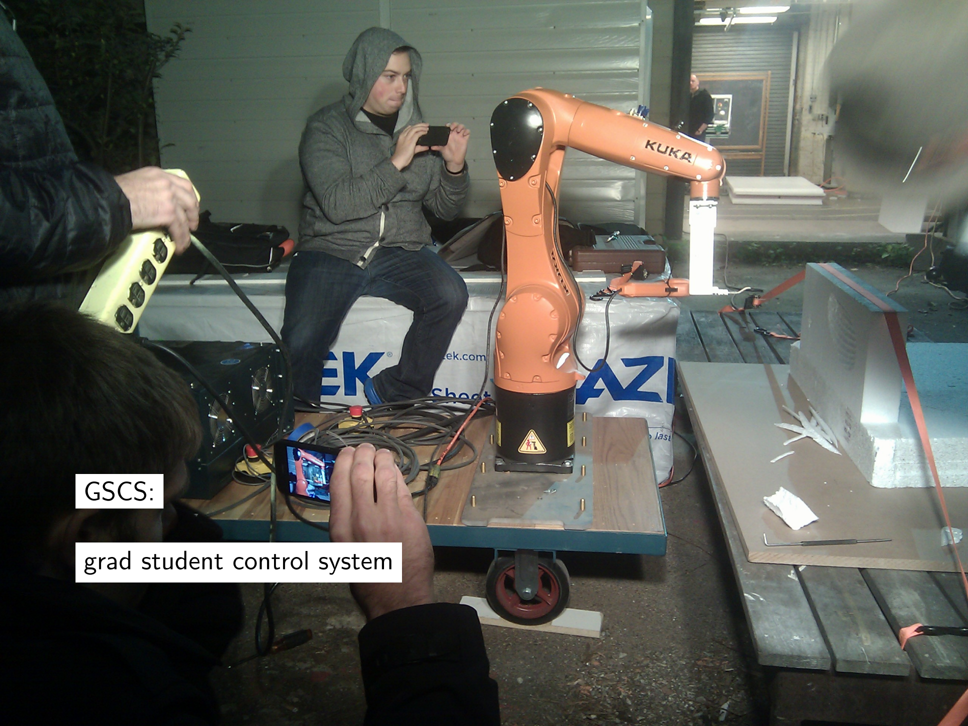


Machines are becoming more affordable

which makes them more accessible

What about less ordinary machines?

CT scanner, 1kW laser, wire EDM, 5-axis mill

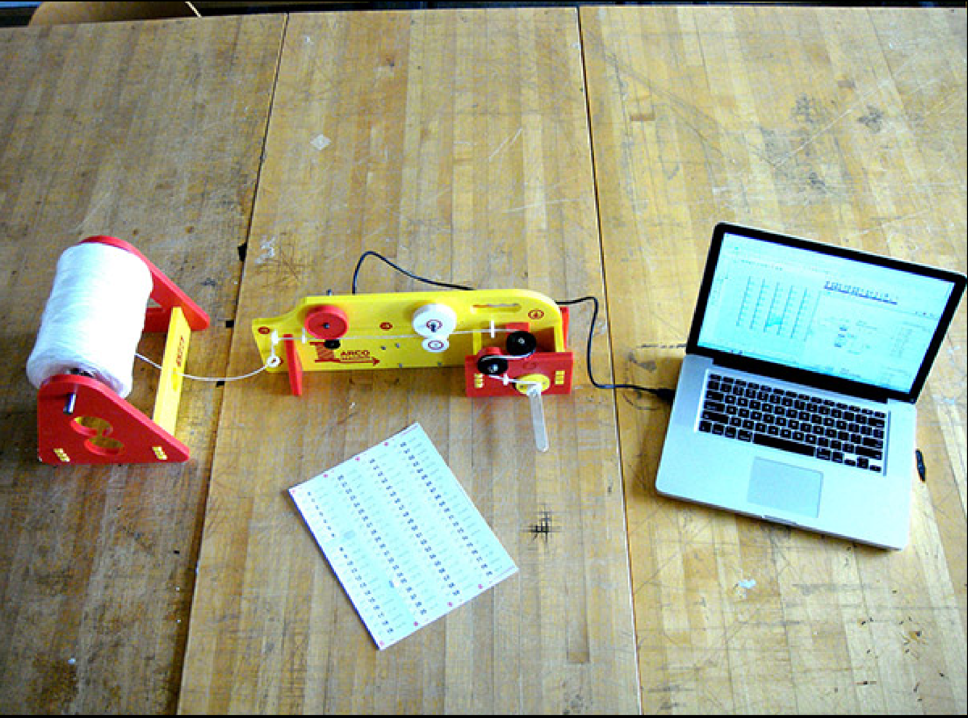


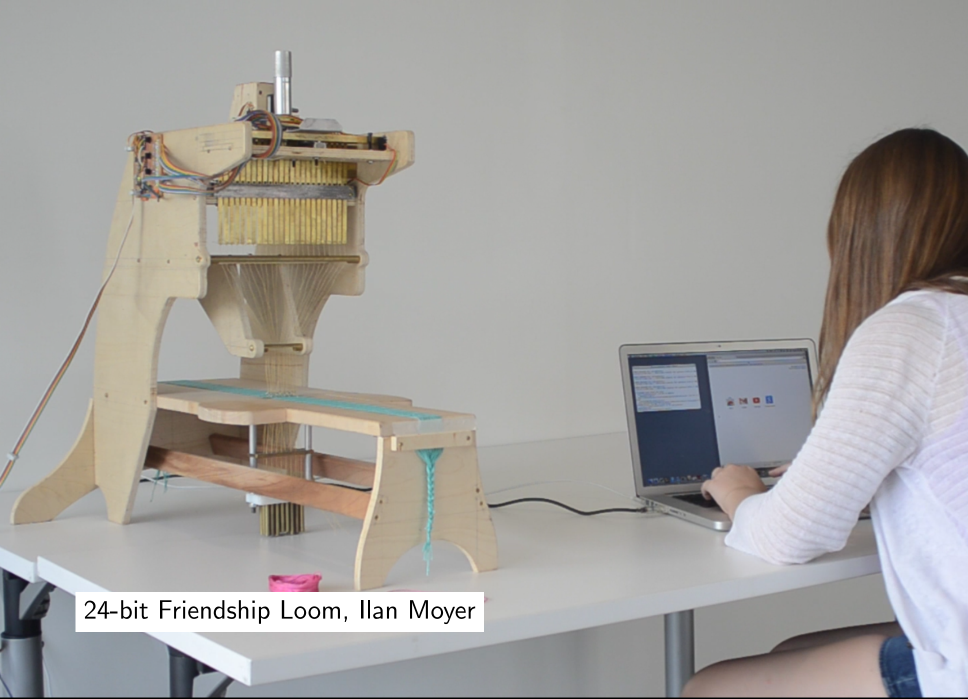
GSCS:

grad student control system

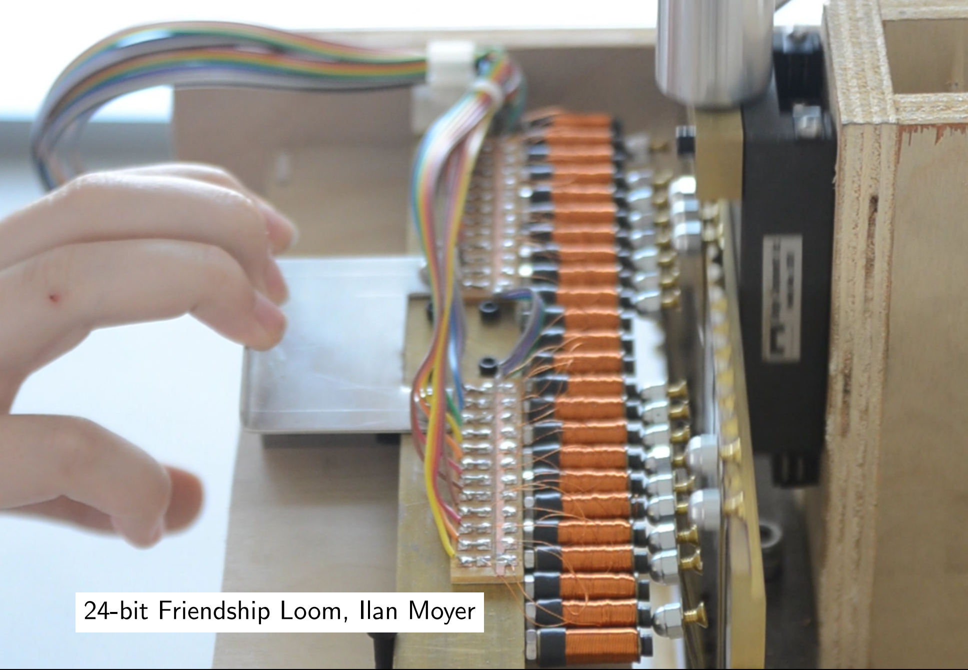
We want a playful system for illegitimate machine offspring.

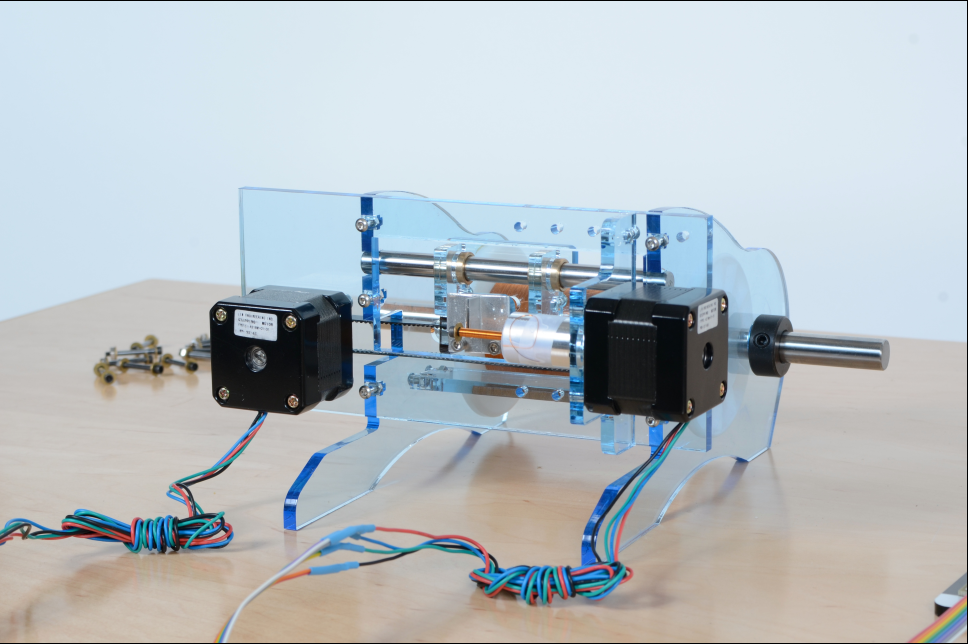
ARCO Madrid ceiling, James Coleman

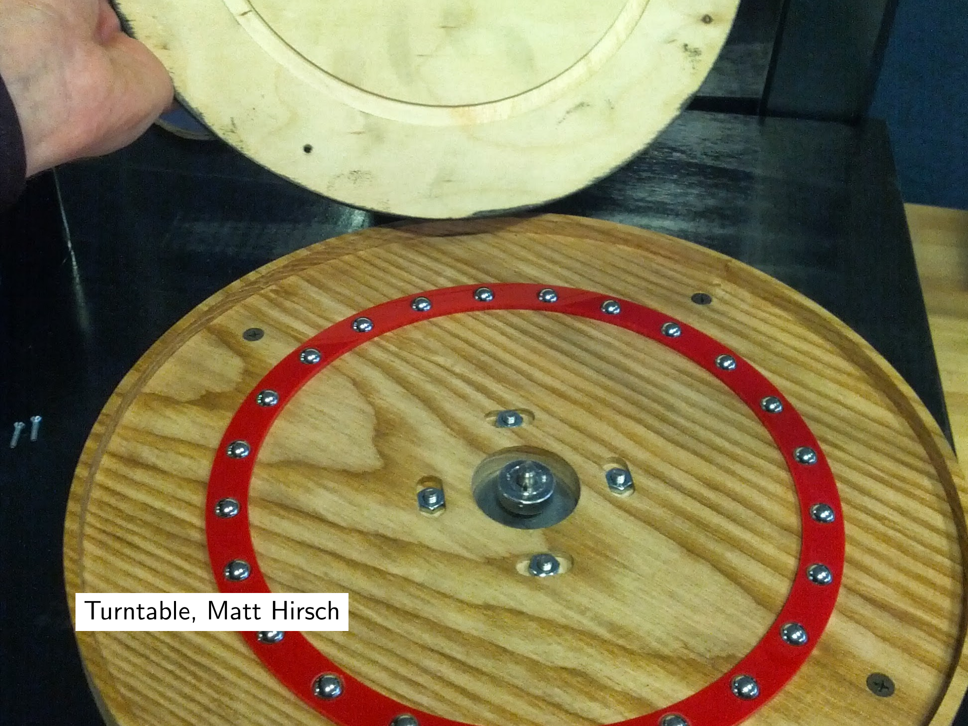


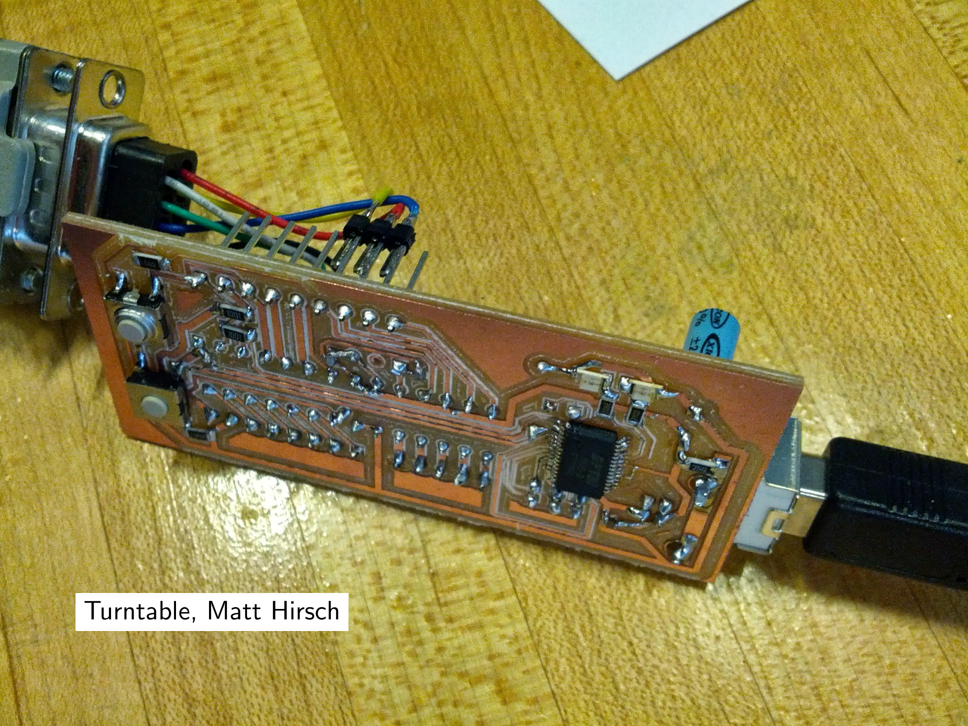


24-bit Friendship Loom, Ilan Moyer


24-bit Friendship Loom, Ilan Moyer

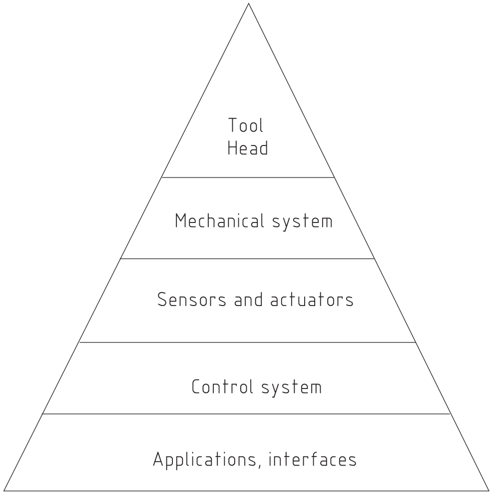



Turntable, Matt Hirsch



Turntable, Matt Hirsch

How can we make it easier
to make machines that make?


Tool
Head
Mechanical system
Sensors and actuators
Control system
Applications, interfaces

By Neil Gershenfeld,
Raffi Krikorian and Danny Cohen

In Barcelona about a century ago, Antoni Gaudí pioneered a fluid building style that seamlessly integrated visual and structural design. The expressive curves of his buildings were not just ornamental facades but also integral parts of the load-bearing structure. Unfortunately, a similar unification has yet to happen for the electronic infrastructure in a building. Switches, sockets and thermostats are grafted on as afterthoughts to the architecture, with functions fixed by buried wiring. Appliances and computers arrive as after-the-fact intrusions. Almost nothing talks to anything else, as evidenced by the number of devices in a typical house or office with differing opinions as to the time of day.

These inconveniences have surprisingly broad implications for construction economics, energy efficiency, architectural expression and, ultimately, quality of life. In the U.S., building buildings is a $1-trillion industry. Of that, billions are spent annually on drawing wiring diagrams, then following, fixing and revising them. Over the years, countless "smart home" projects have sought to find new applications for intelligent building infrastructure—neglecting the enormous existing demand for facilities that can be programmed by their occupants rather than requiring contractors to fix their functionality in advance.

Any effort to meet that demand, though, will be doomed if a lightbulb requires a skilled network engineer to install it and the services of a corporate IT department to manage it. The challenge of improving connectivity requires neither gigabit speeds nor gigabyte storage but rather the opposite: dramatic reductions in the cost and complexity of network installation and configuration.

Over the years, a bewildering variety of standards have been developed to interconnect household devices, including X10, LonWorks, CEBus, BACnet, ZigBee, Bluetooth, IrDA and HomePlug. The situation is analogous to that in the 1960s when the Arpanet, the Internet's predecessor, was developed. There were multiple types of computers and networks then, requiring special-purpose hardware to bridge these islands of incompatibility.

The solution to building a global network out of heterogeneous local networks, called internetworking, was found in two big ideas. The first was packet switching: data are chopped up into packets that can be routed independently as needed and then recombined. This technique marked a break from the traditional approach, used in telephone networks, of dedicating a static circuit to each connection. The second idea was the "end-to-end" principle: the behavior of the network should be determined by what is connected to it rather than by its internal construction, a concept embodied in the Internet Protocol (IP). Gradually the Internet expanded to handle applications ranging from remote computer access to e-commerce to interactive video. Each of these services introduced new types of data for packets to carry, but engineers did not need to change the network's hardware or software to implement them.

These principles have carried the Internet through three decades of growth spanning seven orders of magnitude in both performance and size—from the Arpanet's 64 sites to today's 200 million registered hosts. They represent timeless insights into good system design, and, crucially, they contain no specific performance requirements. With great effort and discipline, technology-dependent parameters were kept out of the specifications so that hardware could evolve without requiring a revision of the Internet's basic architecture.

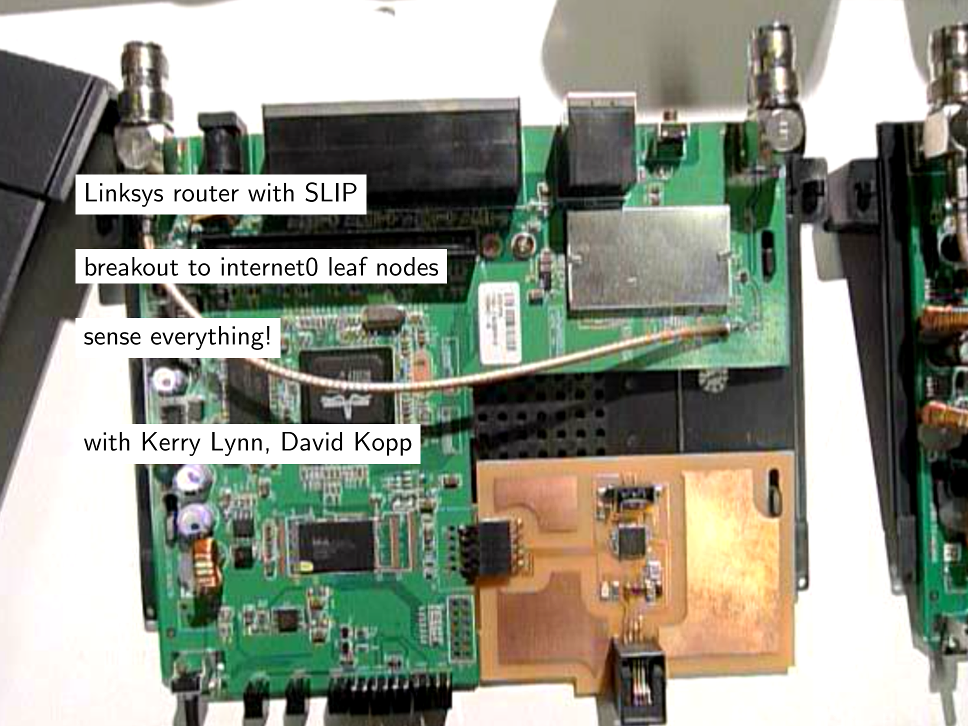
These same ideas can now solve the problem of connecting

The Internet of Things

The principles that gave rise to the Internet are now leading to a new kind of network of everyday devices, an "Internet-0"

EVEN SOMETHING AS SIMPLE as a lightbulb could ... onnected directly to the Internet, ... quipped with cheap circuitry that ... gnals along the electrical wiring.

Scientific American 2004

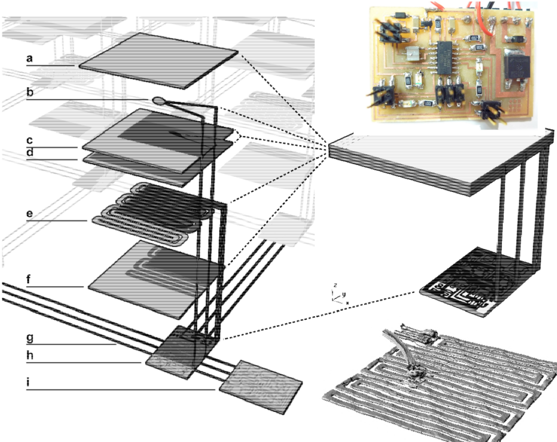


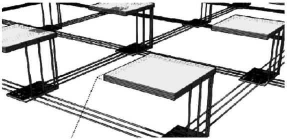
Linksys router with SLIP

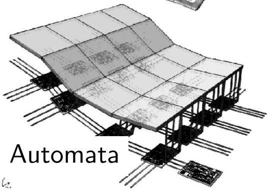
breakout to internet0 leaf nodes

sense everything!

with Kerry Lynn, David Kopp

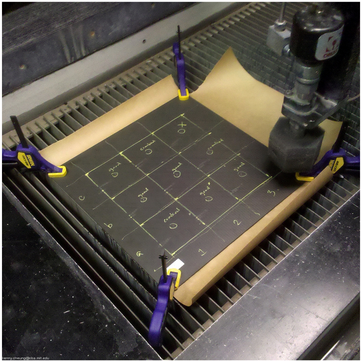


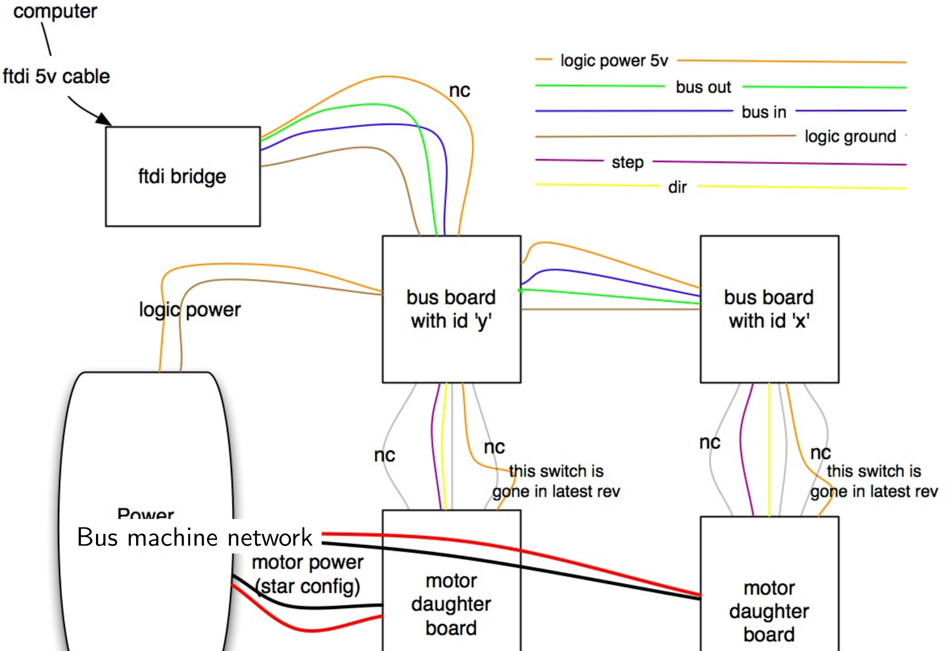




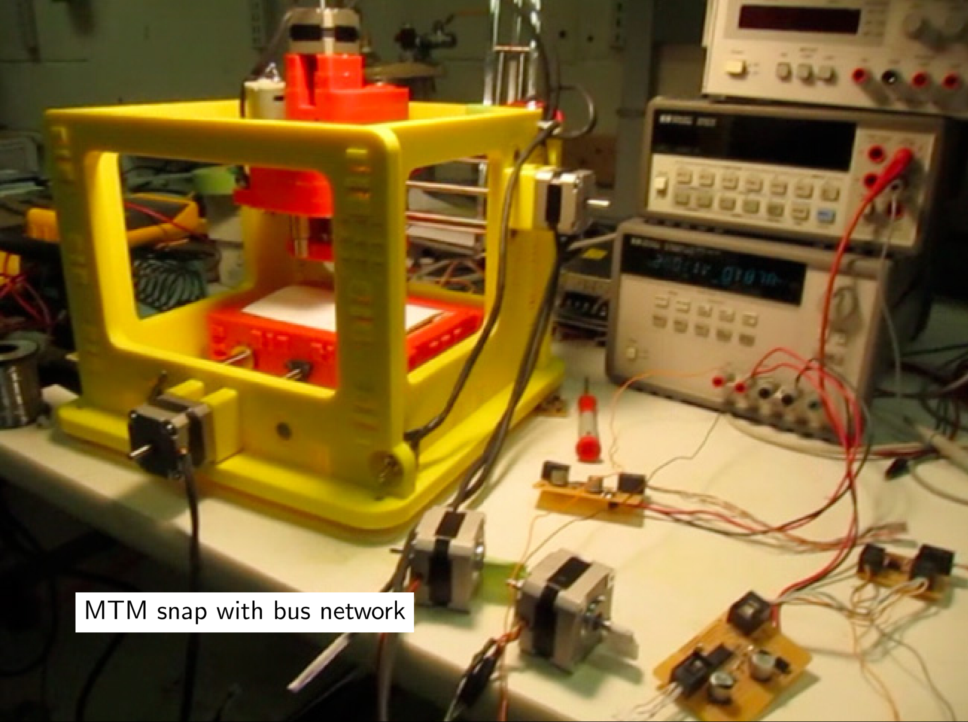
a,b,c,d, # Asynchronous Packet Automata

i





Bus machine network

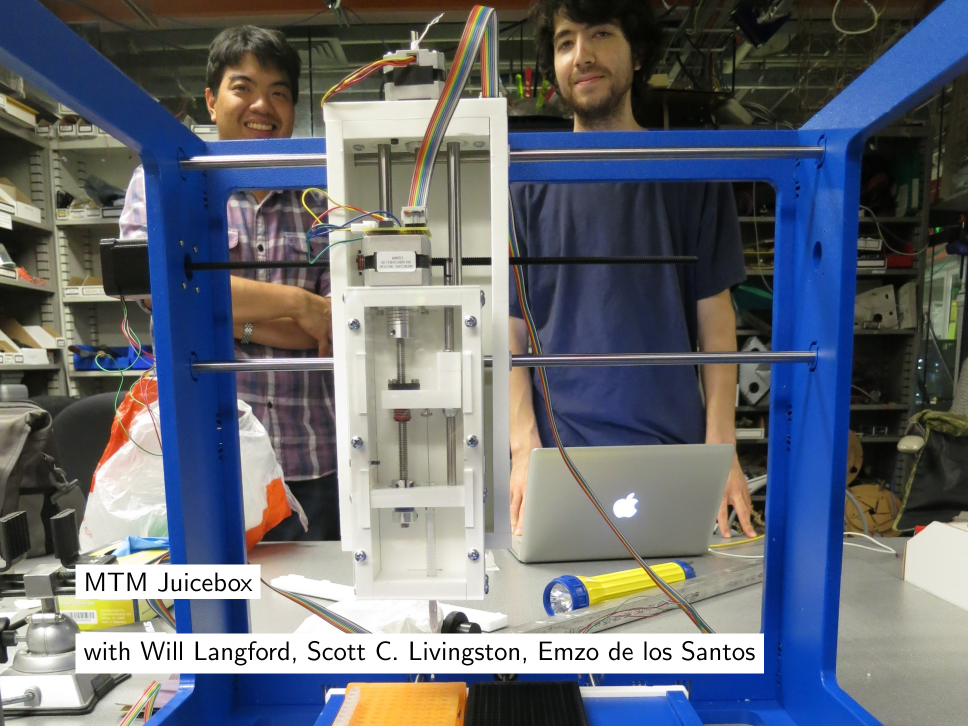


MTM snap with bus network

```
#------VIRTUAL MACHINE------
class virtualMachine(machines.virtualMachine):

	def initInterfaces(self):
		if self.providedInterface: self.fabnet = self.providedInterface		#provid
		else: self.fabnet = interfaces.gestaltInterface('FABNET', interfaces.serialInte

	def initControllers(self):
		self.xAxisNode = nodes.networkedGestaltNode('X Axis', self.fabnet, filename = '
		self.yAxisNode = nodes.networkedGestaltNode('Y Axis', self.fabnet, filename = '
		self.zAxisNode = nodes.networkedGestaltNode('Z Axis', self.fabnet, filename = '
		self.xyzNode = nodes.compoundNode(self.xAxisNode, self.yAxisNode, self.zAxisNod

	def initCoordinates(self):
		self.position = state.coordinate(['mm','mm','mm'])	#X,Y,Z

	def initKinematics(self):
		self.xAxis = elements.elementChain.forward([elements.microstep.forward(4), elem
		self.yAxis = elements.elementChain.forward([elements.microstep.forward(4), elem
		self.zAxis = elements.elementChain.forward([elements.microstep.forward(4), elem

		self.stageKinematics = kinematics.direct(3)	#direct drive on all three axes

	def initFunctions(self):
		self.move = functions.move(virtualMachine = self, virtualNode = self.xyzNode, a
')
		self.jog = functions.jog(self.move)	#an incremental wrapper for the move fu
		pass
```

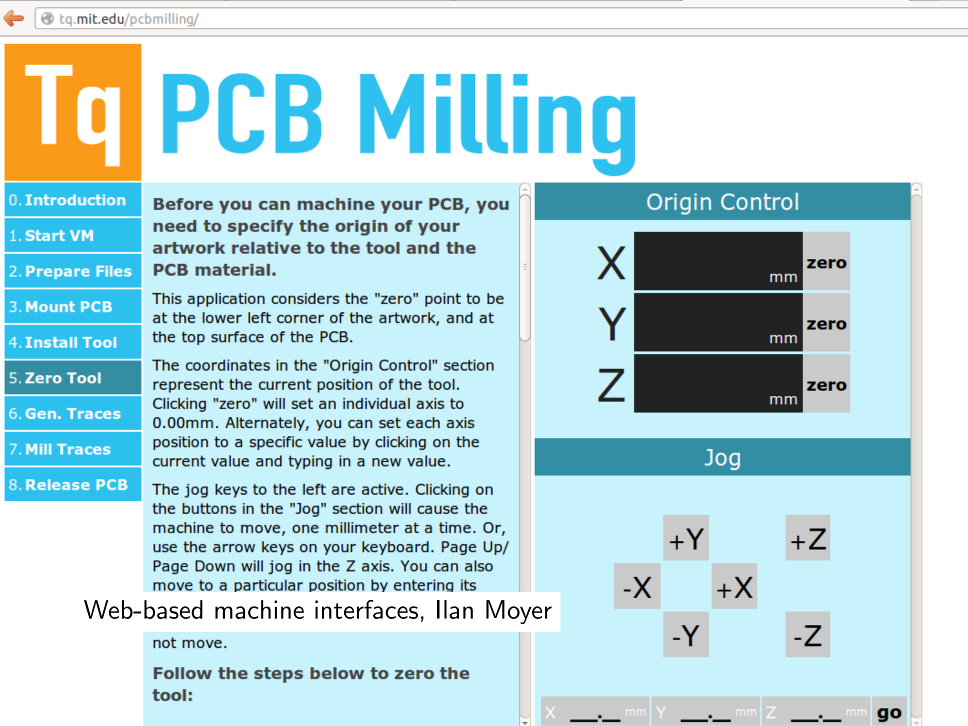
Initializing a MTM snap in pyGestalt

Adding an arbitrary RS-232 interface as a pyGestalt Node

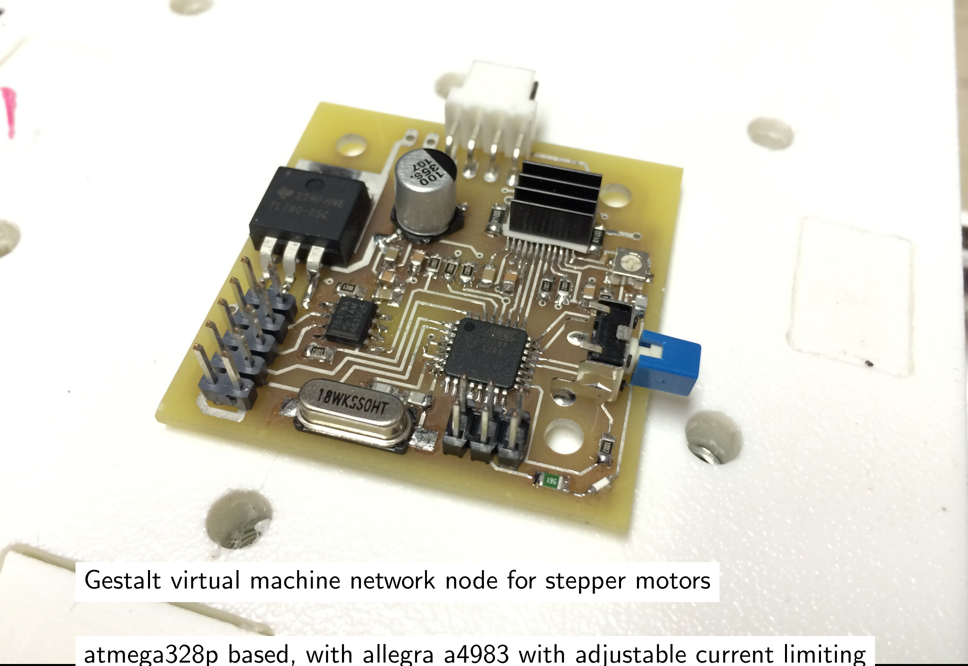


MTM Juicebox

with Will Langford, Scott C. Livingston, Emzo de los Santos

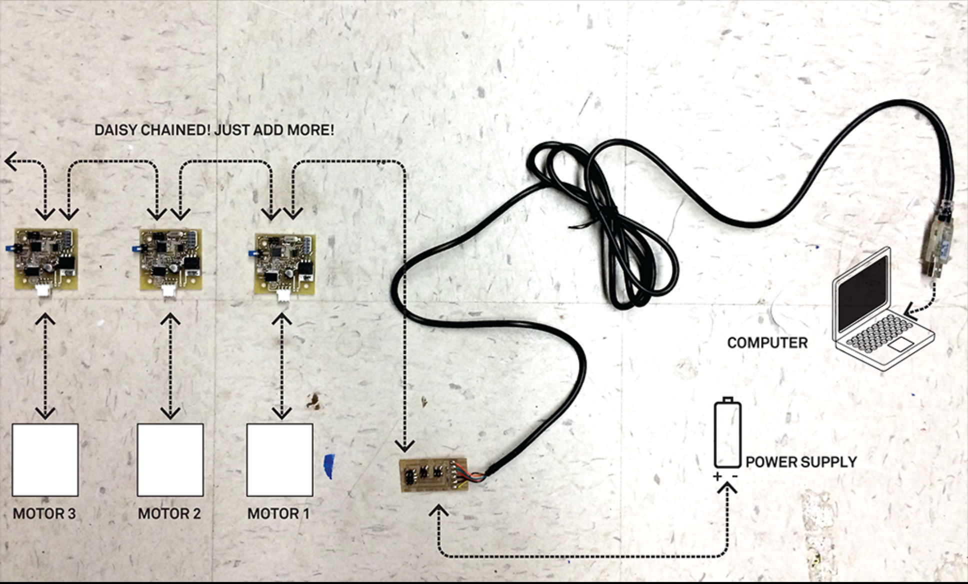


Web-based machine interfaces, Ilan Moyer



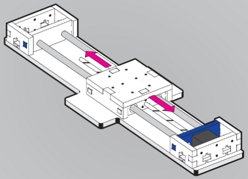
Gestalt virtual machine network node for stepper motors

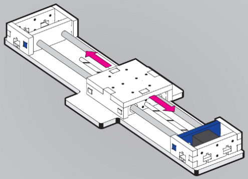
atmega328p based, with allegra a4983 with adjustable current limiting

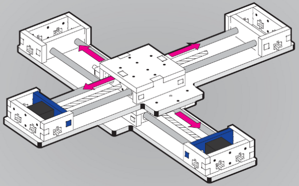


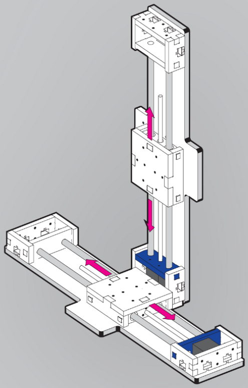
Gestalt virtual machine network

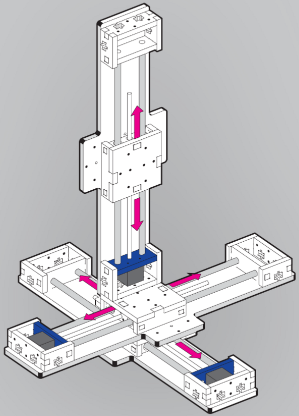
Can the rest be
more modular too?

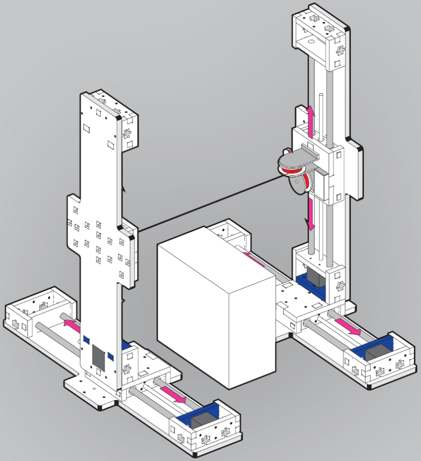


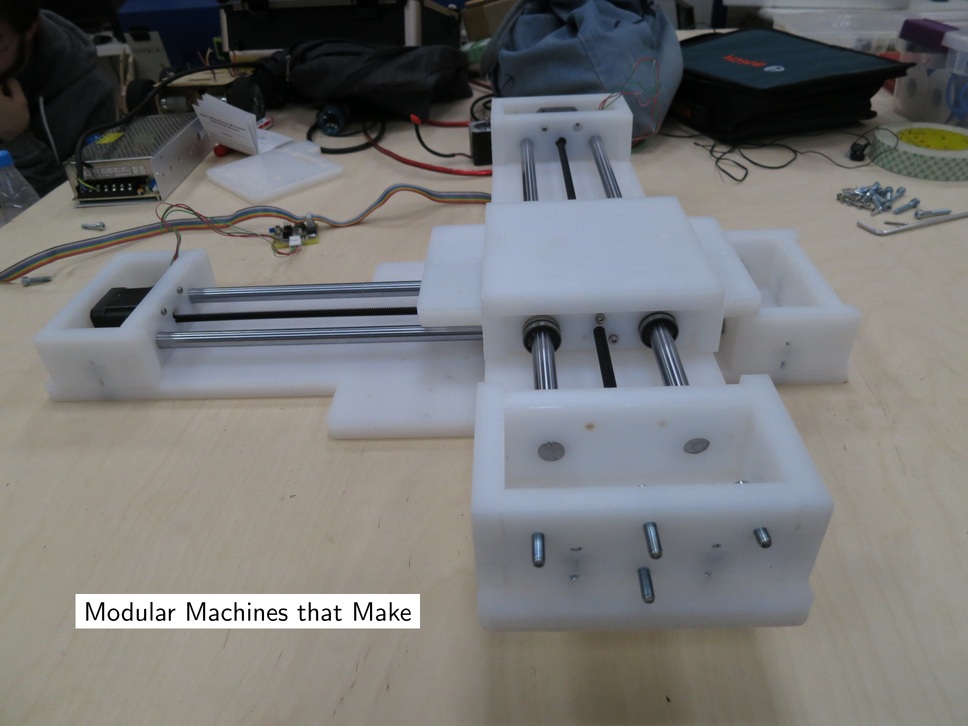




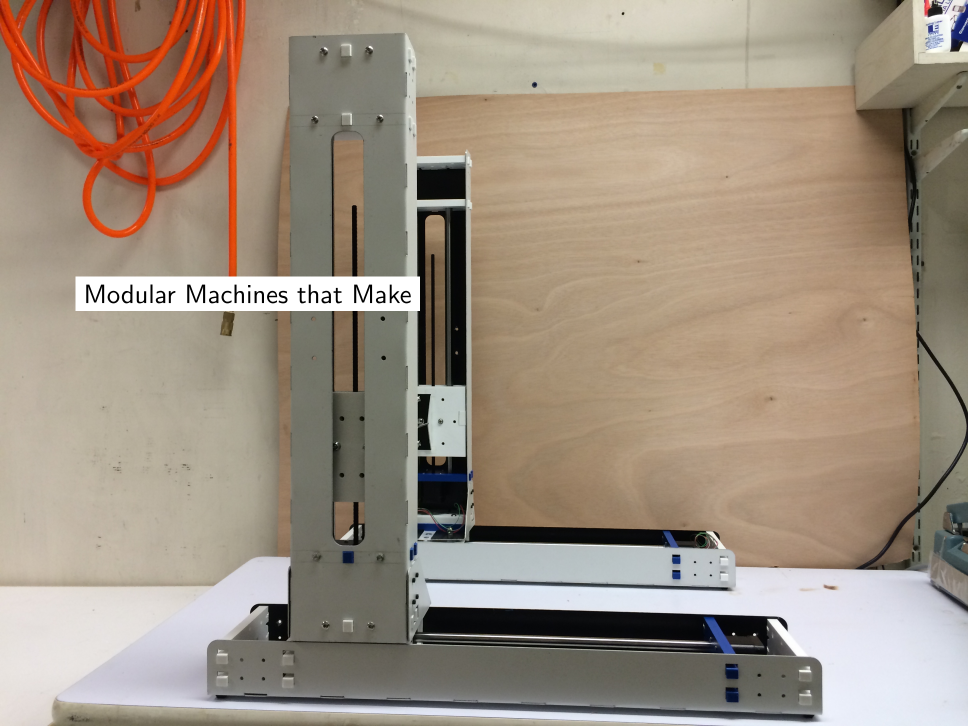


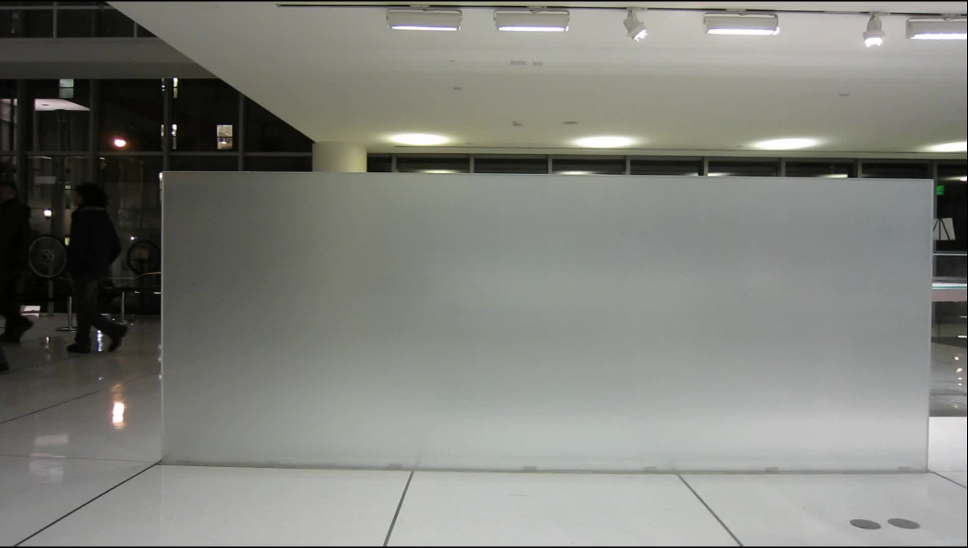





Modular Machines that Make

Modular Machines that Make


Modular Machines that Make



Rapid prototyping of

rapid prototyping machines

mtm.cba.mit.edu

thank you

CCC and fbz

Ilan Moyer

Jonathan Ward

David Mellis

James Coleman

Matt Hirsch

Ben Peters