The Digital Revolution: Machines That Make

Ilan Moyer – CTMTM Co-Founder
Natan Linder – MIT Media Lab
Agenda

• Digital FAB Overview
• CBA MTM Project
• How make something with MTMs
From CNC to FABLAB

Ivan Sutherland, 1963
Sketchpad

William Pease, 1950
“Card-a-matic Milling Machine”
Rapid prototyping?

- Rapid prototyping (industrial design)
- Scalability (parametric designs)
- Object (or manufacturing) for one person
Rapid prototyping?

Progress

Latent Effort

Effort

Progress is a state change.

Rapid Prototyping is closed loop.
Rapid prototyping?

- Software Engineering
- Electrical Engineering
- Mechanical Engineering
Techniques

Additive

Subtractive
Additive

- **Selective laser sintering (SLS):** Thermoplastics, metals, sand
- **Fused Deposition Modeling (FDM):** Thermoplastics
- **Stereolithography (SL):** Photopolymer
- **Electron Beam Melting (EBM):** Titanium alloys
- **3D Ceramic Printing:** Various clay and ceramic materials
Personal Fabrication & DIY

- RepRap
- Fab@Home
- MakerBot Industries
Parametric XY Stage
PCB Mill
Foamcore CNC
Plaster Disaster
Plaster Disaster
Made in India
MTM

- Multifab machines – multi-functions
- Open source
- Networked
- Low cost
- Exploring different materials and form factors
Mantis < $100! – David Carr

http://makeyourbot.org/
Vinyl Cutter
Auto-Pipetter
Auto-Pipetter
Networked motor control
Spindle control
Virtual Machines Control

```python
# Example code snippet

mtm.machine_position = [float(0), float(0), float(0)]
mtm.initialize_debug("apmmcbborder.py")
traverse_speed = 30
retract_speed = 2.0
cutting_speed = 10
plunge_speed = 2.0
z_down = -0.08
z_up = 0.05
mtm.spindle_on(1)
mtm.backlash_compensation_on()
for i in range(3):
    z_down = -0.08*(float(i+1)/3.0)

mtm.move(z = z_up, rate = retract_speed)
mtm.move(z = z_down, rate = plunge_speed)
mtm.move(z = z_up, rate = traverse_speed)
mtm.move(z = z_down, rate = cutting_speed)
...

mtm.move(z = z_up, rate = retract_speed)
mtm.move(z = z_down, rate = plunge_speed)
mtm.move(z = z_up, rate = traverse_speed)
mtm.move(z = z_down, rate = cutting_speed)
mtm.move(z = z_up, rate = retract_speed)
mtm.move(z = z_down, rate = plunge_speed)
mtm.move(z = z_up, rate = traverse_speed)
mtm.move(z = z_down, rate = cutting_speed)
mtm.move(z = z_up, rate = retract_speed)
mtm.move(z = z_down, rate = plunge_speed)
mtm.move(z = z_up, rate = traverse_speed)
mtm.move(z = z_down, rate = cutting_speed)
mtm.move(z = z_up, rate = retract_speed)
mtm.move(z = z_down, rate = plunge_speed)
mtm.move(z = z_up, rate = traverse_speed)
mtm.move(z = z_down, rate = cutting_speed)
mtm.move(z = z_up, rate = retract_speed)
mtm.move(z = z_down, rate = plunge_speed)
mtm.move(z = z_up, rate = traverse_speed)
mtm.move(z = z_down, rate = cutting_speed)
mtm.move(z = z_up, rate = retract_speed)
mtm.move(z = z_down, rate = plunge_speed)
mtm.move(z = z_up, rate = traverse_speed)
mtm.move(z = z_down, rate = cutting_speed)
mtm.move(z = z_up, rate = retract_speed)
mtm.move(z = z_down, rate = plunge_speed)
mtm.move(z = z_up, rate = traverse_speed)
mtm.move(z = z_down, rate = cutting_speed)
mtm.move(z = z_up, rate = retract_speed)
mtm.move(z = z_down, rate = plunge_speed)
mtm.move(z = z_up, rate = traverse_speed)
mtm.move(z = z_down, rate = cutting_speed)
mtm.move(z = z_up, rate = retract_speed)
mtm.move(z = z_down, rate = plunge_speed)
mtm.move(z = z_up, rate = traverse_speed)
mtm.move(z = z_down, rate = cutting_speed)
```

**VIRTUAL MACHINE ENVIRONMENT 10 - REV. A2**

```
------------ MACHINE COMPONENTS ------------

mtm()

--> XYZ Driver is LAUREL TRI-AXIS @ 192.168.233.146
--> Spindle is LAUREL TRI-AXIS @ 192.168.166.80
--> Trunnion is GENERIC VMO @ ?..?..?

------------ EXECUTE SINGLE COMMAND -----------

COMMAND: mtm.spindle_on(1)

------------ LOAD AND RUN VMC FILE -----------

FILE: apmmcbborder.py run as lines run as file

CURRENT LINE: mtm.spindle_on(0)
```
5 Axis Trunnion
Overview
MTM "Little John" is a multi-purpose mid-size rapid prototyping machine that which has the goal of being a personal fabricator capable of performing a variety of tasks (3D printing, milling, scanning, vinyl cutting, etc.) at a price point in the hundreds rather than thousands of dollars.

All the MTM-LJ parts are currently using two 4’ x 4’ x 1” sheets. Parts layout is using spacing for cutting with a 0.25” straight end-mill. Pilot holes should be cut first using a 0.125” straight end-mill, see Fabrication section for detailed screenshots. To make tool-path generation the DXF & Rhino files below are using layers.
Specifications / Key Features:
Dimensions: 2’x2’x14.5”
Working envelope: 9”x18”x5.5”
Interchangeable tool interface, multiple heads
10 Networked motor control (unipolar+bipolar support)
Open-source H/W + S/W Linux RT + EMC2
Multifab Toolinterface

MTM Little John tool interface is designed to support multiple tools-heads. Example tools can be a combination of plastic extruders, printing/plotting, milling, scanning.
Plastruder
How make something with MTMs
Keychain Flashlight
Top projection → DXF - Rhino
Schematics & PCB Layout – Eagle
Cad.py – Toolpath
Keyfob enclosure toolpath
TIND
G90
F3.0000
S5000.0000
M08
G020-0.2750
M03
G00X0.6970Y2.6130Z0-0.2750
G01Z0.0100
X0.6950Y2.6110
X0.6770Y2.6110
X0.6750Y2.6090
X0.6690Y2.6070
X0.6640Y2.6070
X0.6570Y2.6050
X0.6490Y2.6050
X0.6390Y2.6030
X0.6370Y2.6010
X0.6270Y2.5990
X0.6230Y2.5990
X0.6210Y2.5970
X0.6150Y2.5970
X0.6130Y2.5950
X0.6050Y2.5950
X0.6030Y2.5930
X0.5970Y2.5930
X0.5940Y2.5910
X0.5890Y2.5910
X0.5830Y2.5870
X0.5810Y2.5850
X0.5750Y2.5830
X0.5730Y2.5810
X0.5670Y2.5790
X0.5650Y2.5770
X0.5590Y2.5750
X0.5570Y2.5730
X0.5490Y2.5670
X0.5430Y2.5630
X0.5350Y2.5570
X0.5290Y2.5530
X0.5230Y2.5510
X0.5150Y2.5410
X0.5130Y2.5330
X0.5090Y2.5310
X0.5030Y2.5230
X0.4930Y2.4910
X0.4860Y2.4850
X0.4690Y2.4790
G0
G00
G04