

# **University of Stuttgart**

Visualization Research Center (VISUS)

frank.heyen@visus.uni-stuttgart.de, michael.sedlmair@visus.uni-stuttgart.de

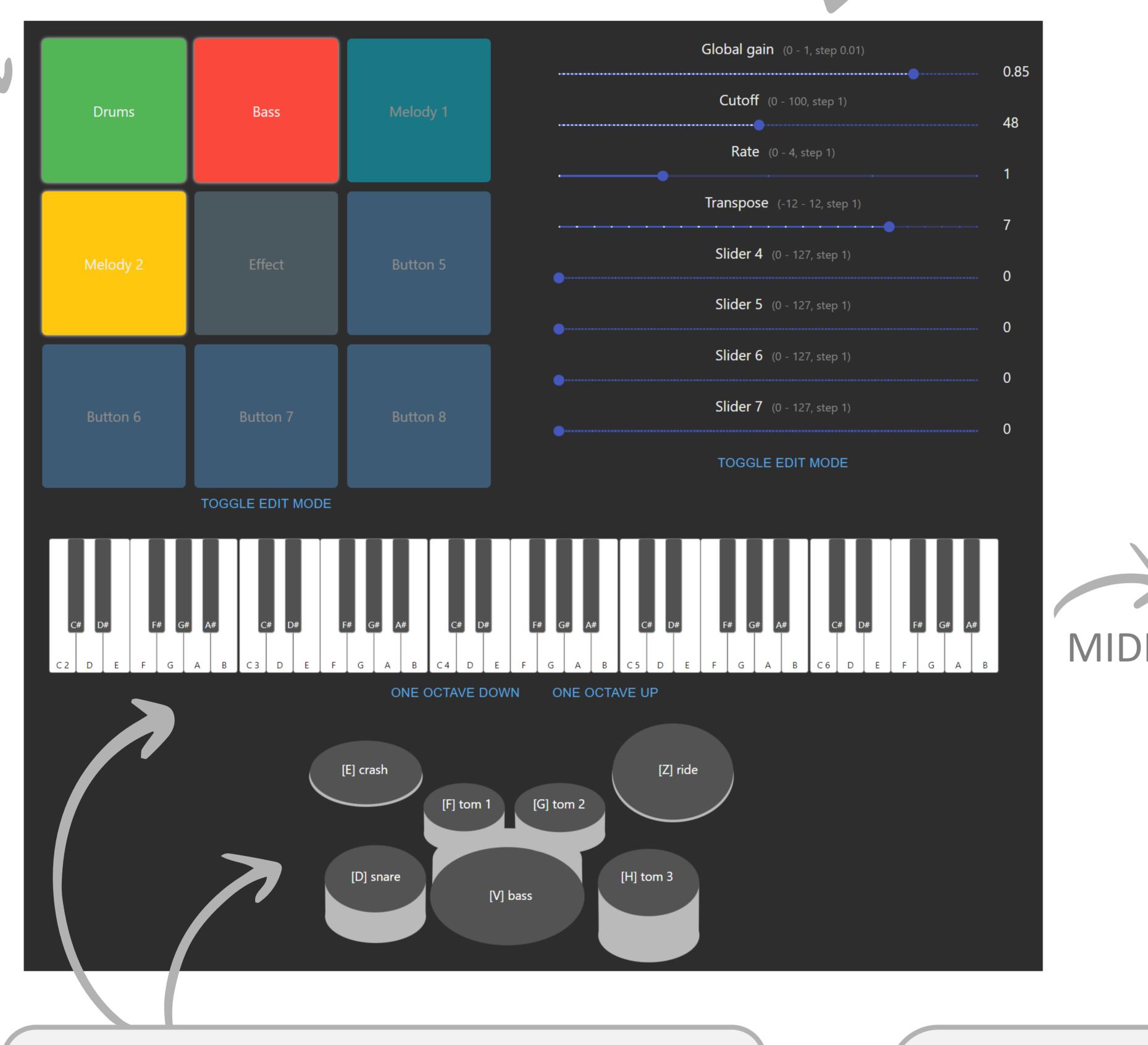
### **Overview**

- Live coding music allows creating music through code with immediate results
- Sonic Pi is a free live-coding environment designed to help teaching children to code
- Some musical features (toggle, fade, improvisation) are hard to implement
- We propose a free and easy to learn visual interface for these features
- The goal is to let users focus on the creative part of live coding
- Compared to hardware, our approach is cheaper and more customizable
- Open-source and accessible: source code and web app available

Frank Heyen, A Web-Based Dilara Aygün, Michael SedImair MIDI Controller for Music Live Coding

#### **Button Grid**

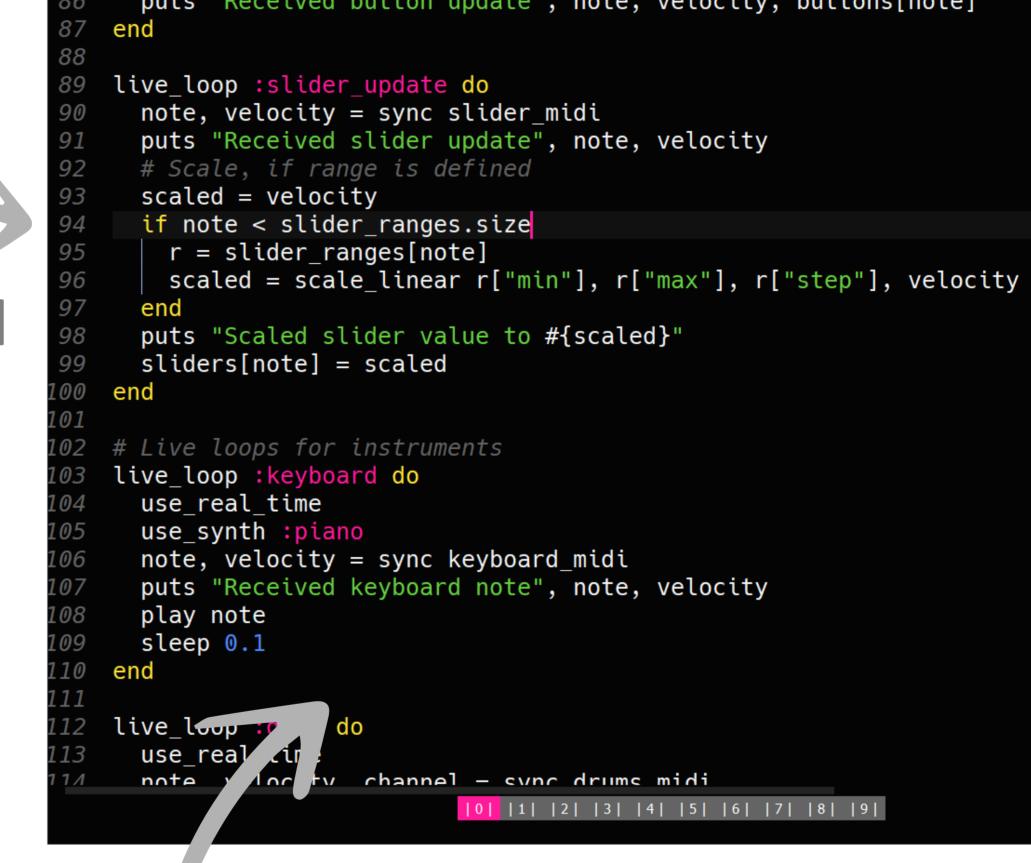
- Allow *toggling* tracks, effects, ...
- Example: toggle a melody line
- Can be customized with colors and labels



#### **Sliders**

- Allow *influencing* effects interactively
- Example: controlling a fading curve while live performing
- Can be customized with minimum, maximum, step values

Sonic Pi	
File Edit Window	
run 🕨 stop 🔳 rec 😑 save 🕂 load ୟ	
65	
66 # Initial values of slider and button values	
67 sliders = $Array.new(100, 0)$	
68 buttons = Array.new(36, false)	
69	
70 # Scales a value from [0, 127] to [min, max] and	
71 <i># rounds to next valid step</i>	
<pre>72 def scale_linear(min, max, step, value)</pre>	
<pre>73 scaled = min + (value / 127.0) * (max - min)</pre>	
74 stepped = scaled / step	
<pre>75 rounded = stepped.round * step</pre>	
76 end	
77	
78 <i># Update loops that keep slider and button states up-to-date</i>	
79 live_loop :button_update do	
80 note, velocity = sync button_midi	
81 if velocity > 0	
82 buttons[note] = true	
83 else	
84   buttons[note] = false	
<pre>85 end 86 puts "Received button update", note, velocity, buttons[note]</pre>	
<pre>86 puts "Received button update", note, velocity, buttons[note] 87 end</pre>	



#### Instruments

- For live improvisation, by interactively inputing notes
- *Keyboard*, played through mouse or touch screen
- Drums, played through mouse or computer keyboard

# **Communication & Boilerplate**

- Communication through *MIDI*
- No technical lock-in: users can switch to hardware MIDI controllers anytime

## **Limitations & Future Work**

- Latency makes improvisation difficult (also for hardware)
- No haptic feedback as with hardware knobs, buttons, sliders
- Future work: integrate visual feedback on what notes are played, allow to visually control effects with ADSR envelopes

- Boilerplate code receives MIDI and changes variables
- Variable values can be used arbitrarily in the user's code



Supplemental material and live demo *qithub.com/visvar/sonic-pi-controller* 



More of our research visvar.github.io



