scadnano

scriptable, web-based port of caDNAno for designing DNA nanostructures

David Doty

University of California, Davis

joint work with Benjamin Lee, Tristan Stérin

FNANO 2020: Foundations of Nanoscience virtual conference, April 2020

University of California, Davis

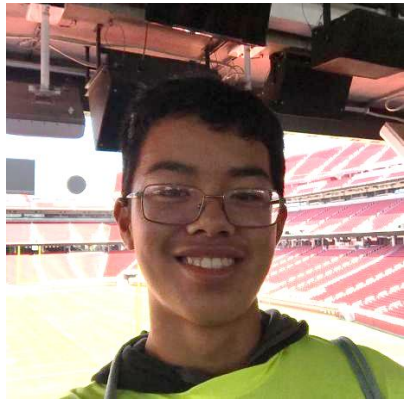
NUI MAYNOOTH
Ollscoil na hÉireann Má Nuad

Hamilton Institute

Acknowledgements

co-developers:

Benjamin Lee



undergraduate CS student, UC-Davis

Tristan Stérin

Ph.D. student, Maynooth University

beta-tester:

Matt Patitz

Associate Professor, University of Arkansas

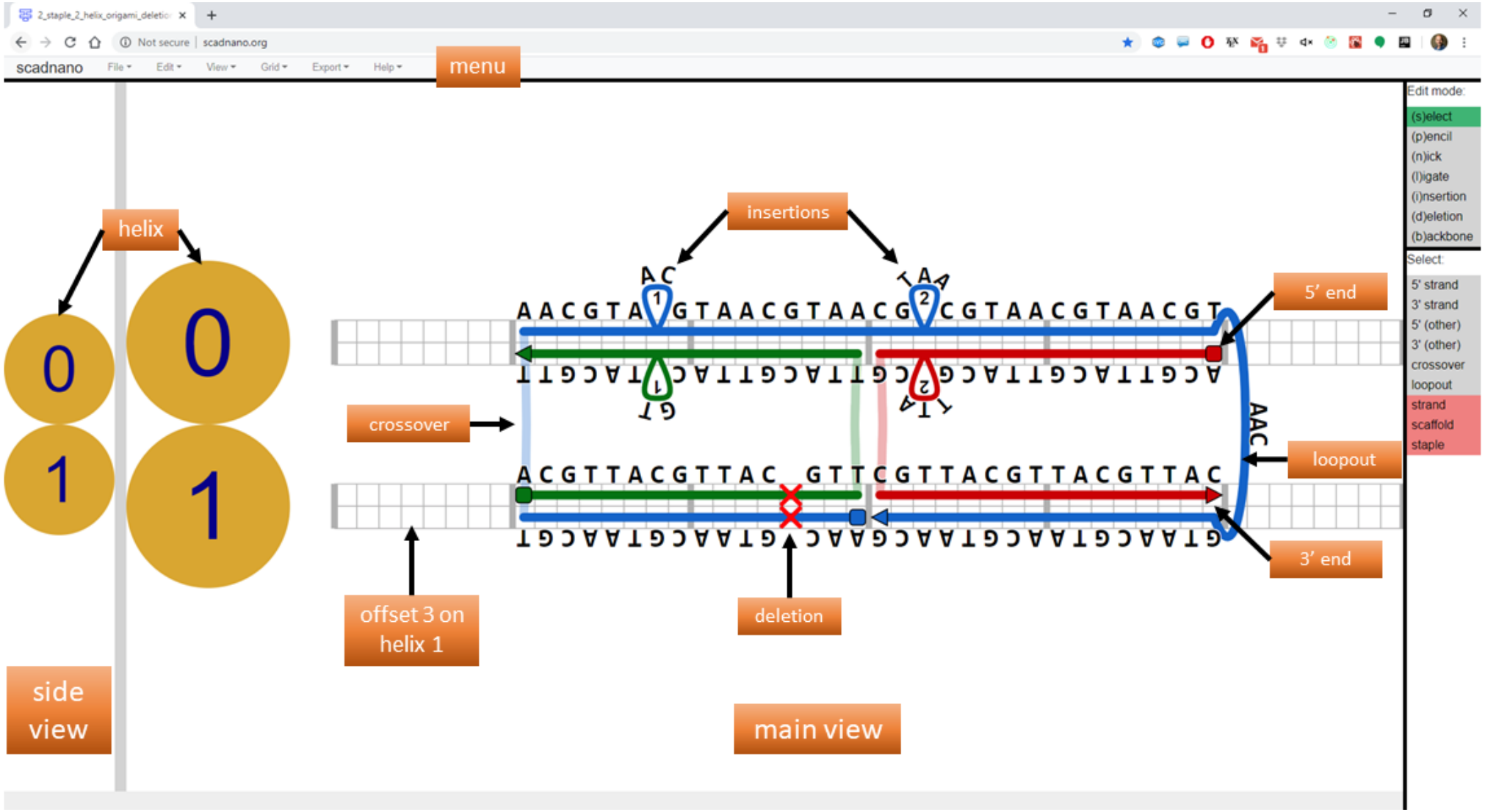
What is scadnano?

- web application: scadnano.org
- *scriptable*-caDNAno
- If you know caDNAno, it's like that, except…
  - browser-based, so no installation necessary
  - scriptable: comes with well-documented scripting library (Python) for creating and modifying DNA structure designs *programmatically*
- If you don't know caDNAno… scadnano is a program to design DNA nanostructures, with an overall goal of being *easy to use*.

Comparison with other DNA design software

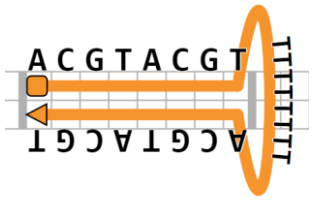
software	main goal	other goals
scadnano	design DNA structures (parallel helices)	ease of installation and use, easy to script, easy to edit manually
cadnano	design DNA structures (parallel helices)	
codenano	design DNA structures (no parallel constraint)	predict 3D structure of DNA from its secondary structure, (only scripting; no manual editing)
Adenita	design DNA structures (no parallel constraint)	modular reuse
ATHENA	design DNA structures (no parallel constraint)	edges connecting parts of structure are rigid, six-helix bundle (6HB) or more compliant, two-helix bundle (2HB or DX)
MrDNA	predict 3D structure of DNA from its secondary structure	predict kinetics
CanDo	predict 3D structure of DNA from its secondary structure	
NUPACK/Vienna RNA	analyze thermodynamic energy of DNA strands	design sequences with prescribed energetics
oxDNA	predict kinetics of DNA by molecular dynamics simulation	

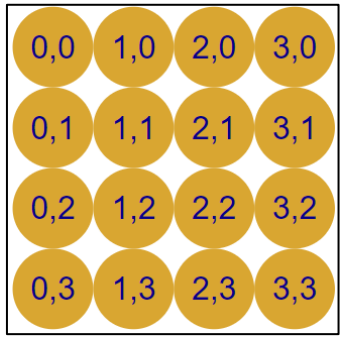
web interface

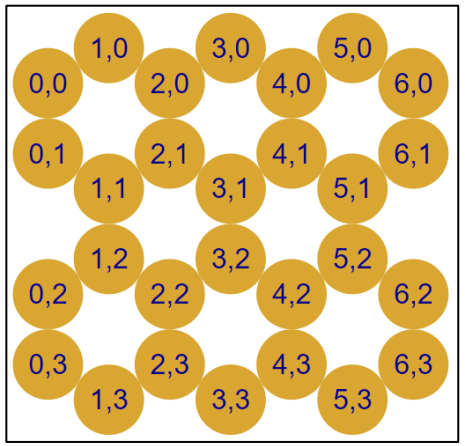


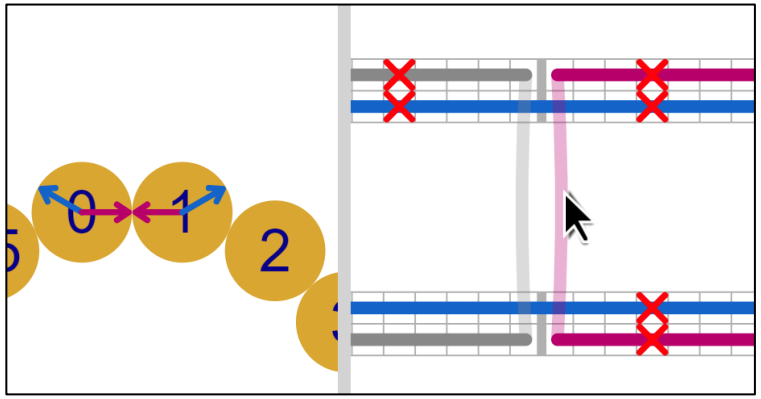
Other features

- features shared with cadnano v2:
  - import/export cadnano files (v. 2/2.2, not yet v. 2.5)
  - export SVG figures
  - export DNA sequences in CSV file
  - square/honeycomb grids
  - visualize backbone angle for help placing crossovers
- scadnano features not found in cadnano v2:
  - export DNA sequences in Excel and text files readable by IDTDNA
  - cut/copy/paste strands
  - saves DNA assignment in file
  - can hide certain helices to help with 3D design
  - loopouts (single-stranded DNA not on any helix)

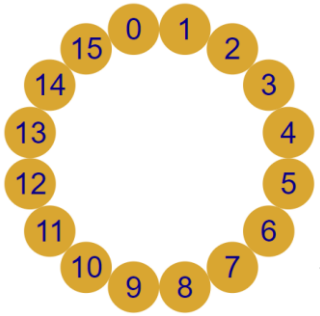




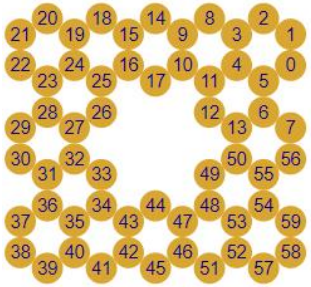


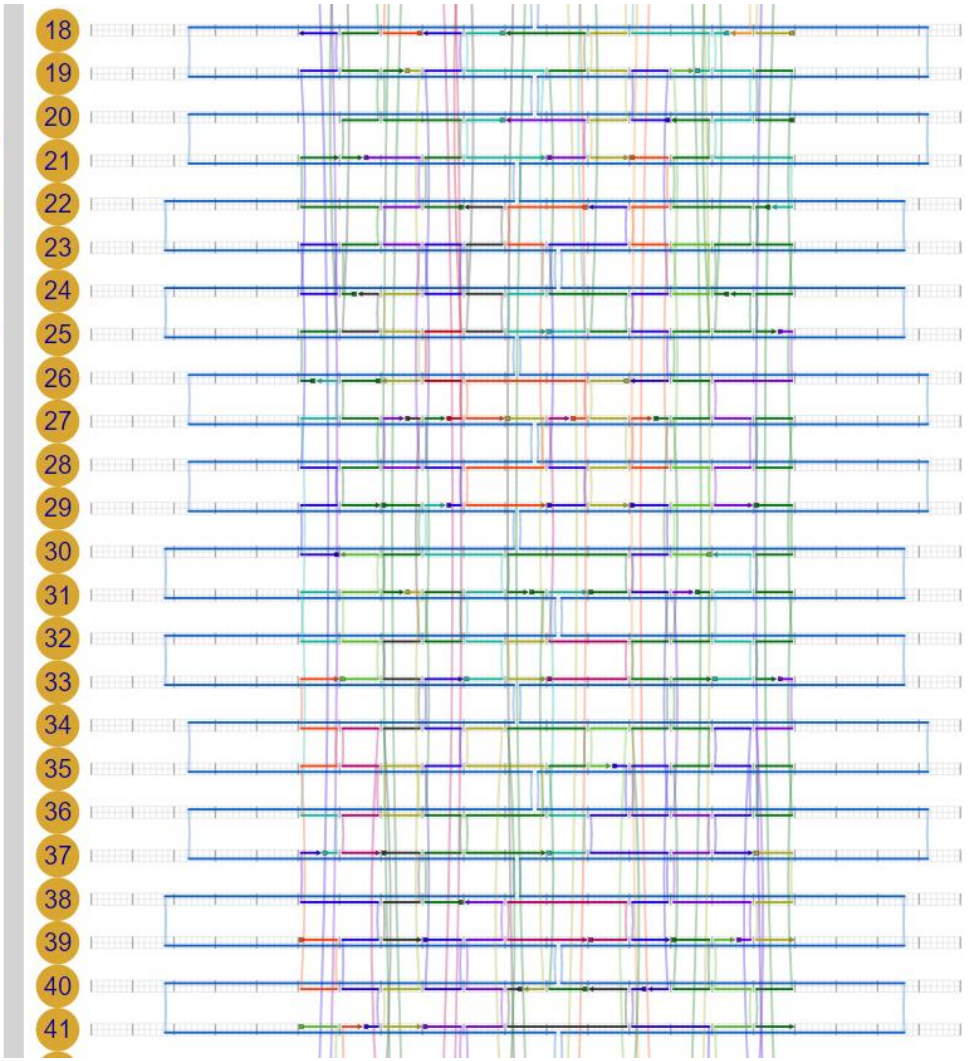


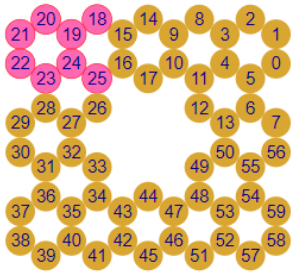
- DNA modifications (e.g., biotin, fluorophore)
- well-documented Python scripting library
- easily-readable file format
- gridless helix placement

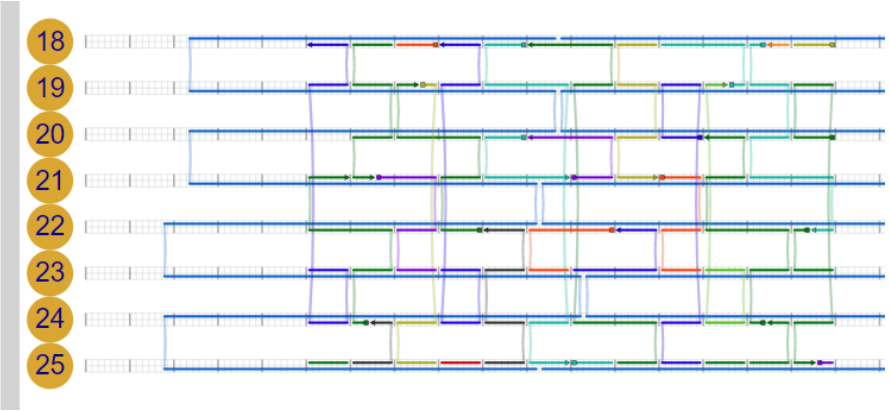


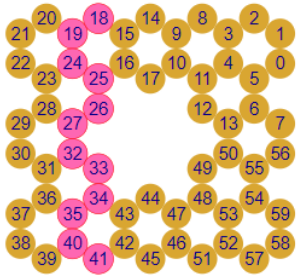
Hiding helices

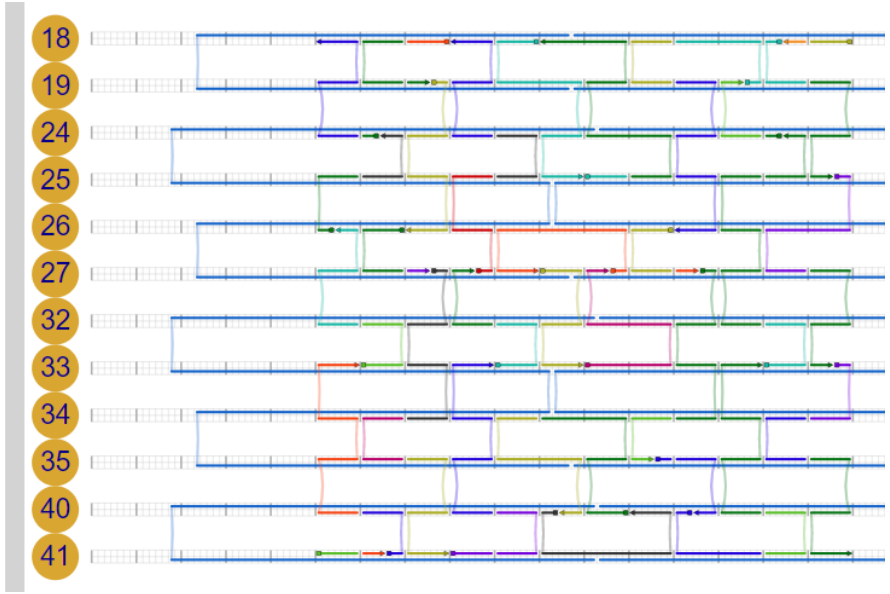






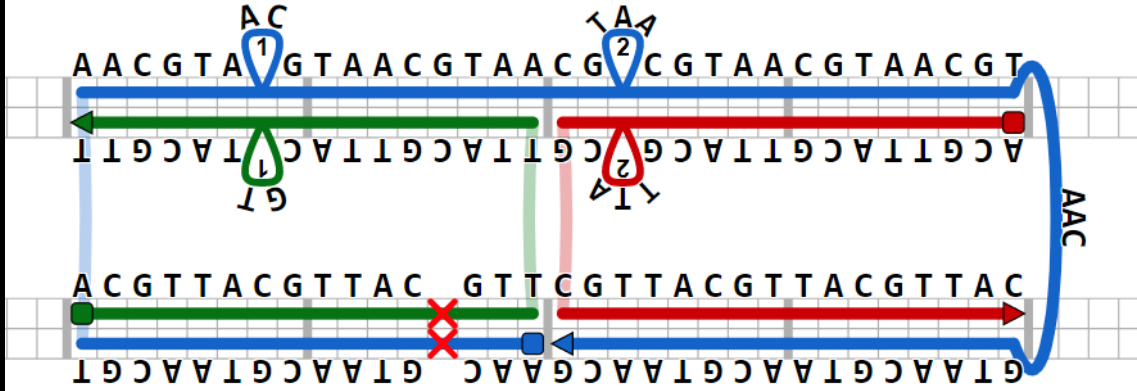






Easily readable file format

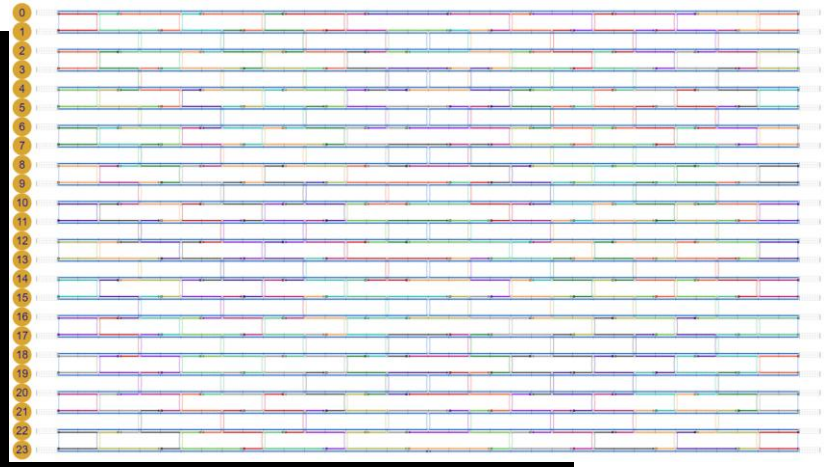

```
{
  "version": "0.3.0",
  "helices": [
    {"max_offset": 48, "grid_position": [0, 0]},
    {"max_offset": 48, "grid_position": [0, 1]}
  ],
  "strands": [
    {
      "color": "#0066cc",
      "dna_sequence": "AACGTAACGTAACGTAACGTAACGTAACGTAACGTAACGTAACGTAACGTAACGTAACGTAACGTAACG",
      "substrands": [
        {"helix": 1, "forward": false, "start": 8, "end": 24, "deletions": [20]},
        {"helix": 0, "forward": true, "start": 8, "end": 40, "insertions": [[14, 1], [26, 2]]},
        {"loopout": 3},
        {"helix": 1, "forward": false, "start": 24, "end": 40}
      ],
      "is_scaffold": true
    },
    {
      "color": "#007200",
      "dna_sequence": "ACGTTACGTTACGTTTTACGTTACGTTACGTT",
      "substrands": [
        {"helix": 1, "forward": true, "start": 8, "end": 24, "deletions": [20]},
        {"helix": 0, "forward": false, "start": 8, "end": 24, "insertions": [[14, 1]]}
      ]
    },
    {
      "color": "#cc0000",
      "dna_sequence": "ACGTTACGTTACGTTACGCGTTACGTTACGTTAC",
      "substrands": [
        {"helix": 0, "forward": false, "start": 24, "end": 40, "insertions": [[26, 2]]},
        {"helix": 1, "forward": true, "start": 24, "end": 40}
      ]
    }
  ]
}
```

helpful when debugging Python scripts

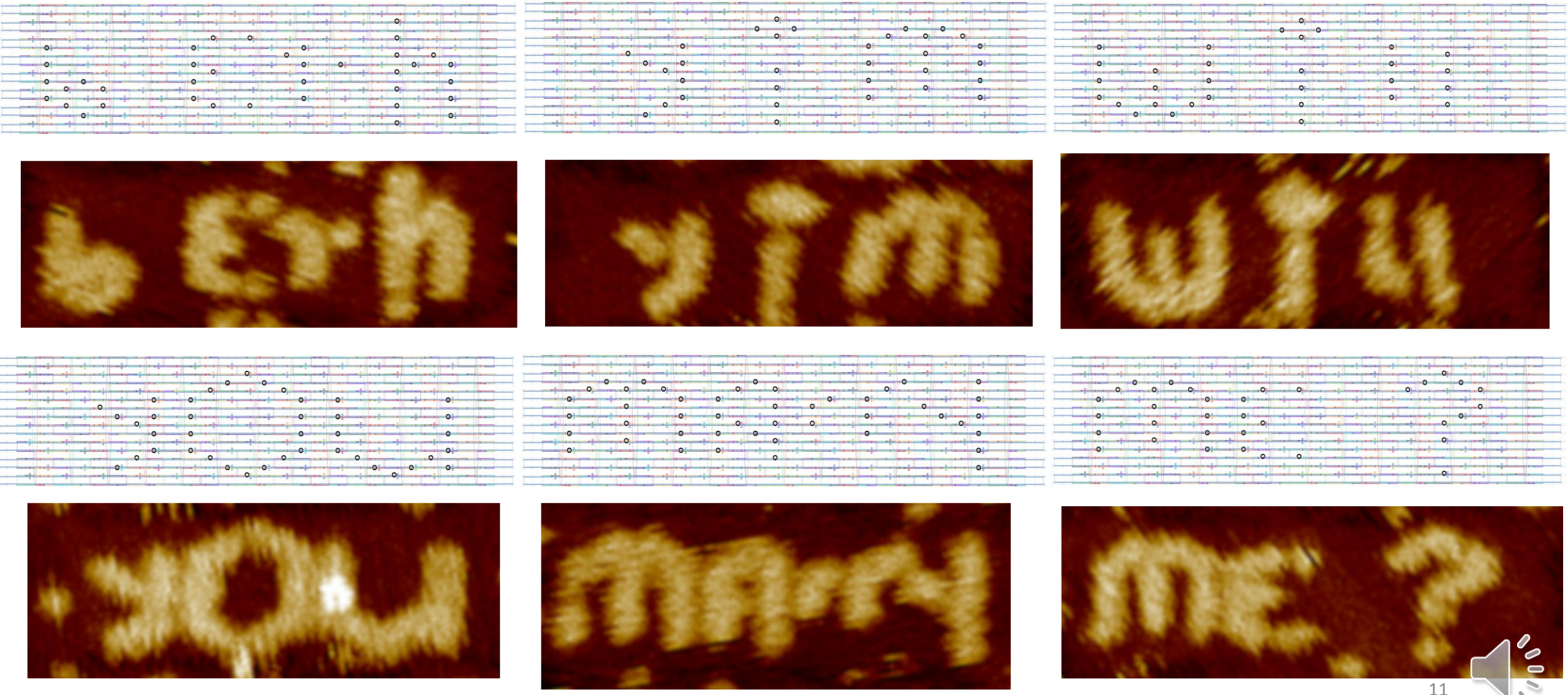
Python scripting library

```python
import scadnano as sc

def main():
    design = precursor_scaffolds()
    add_scaffold_nicks(design)
    add_scaffold_crossovers(design)
    scaffold = design.strands[0]
    scaffold.set_scaffold()
    add_precursor_staples(design)
    add_staple_nicks(design)
    add_staple_crossovers(design)
    design.assign_m13_to_scaffold()
    return design

def add_staple_crossovers(design: sc.DNADesign):
    for helix in range(23):
        start_offset = 24 if helix % 2 == 0 else 40
        for offset in range(start_offset, 296, 32):
            if offset != 152:  # skip crossover near seam
                design.add_full_crossover(helix1=helix, helix2=helix + 1,
                                          offset1=offset, forward1=helix % 2 == 1)

def add_staple_nicks(design: sc.DNADesign):
    for helix in range(24):
        start_offset = 32 if helix % 2 == 0 else 48
        for offset in range(start_offset, 280, 32):
            design.add_nick(helix, offset, forward=helix % 2 == 1)

if __name__ == '__main__':
    design = main()
    design.write_scadnano_file()
```



```python
def precursor_scaffolds() -> sc.DNADesign:
    helices = [sc.Helix(max_offset=304) for _ in range(24)]
    scaffolds = [sc.Strand([sc.Substrand(helix=helix,
                  forward=helix % 2 == 0, start=8, end=296)])
                  for helix in range(24)]
    return sc.DNADesign(helices=helices, strands=scaffolds,
                        grid=sc.square)

def add_scaffold_nicks(design: sc.DNADesign):
    for helix in range(1, 24):
        design.add_nick(helix=helix, offset=152,
                        forward=helix % 2 == 0)

def add_scaffold_crossovers(design: sc.DNADesign):
    crossovers = []

    # scaffold interior
    for helix in range(1, 23, 2):
        crossovers.append(sc.Crossover(helix1=helix, helix2=helix + 1,
                                       offset1=152, forward1=False))

    # scaffold edges
    for helix in range(0, 23, 2):
        crossovers.append(sc.Crossover(helix1=helix, helix2=helix + 1,
                          offset1=8, forward1=True, half=True))
        crossovers.append(
            sc.Crossover(helix1=helix, helix2=helix + 1, offset1=295,
                         forward1=True, half=True))

    design.add_crossovers(crossovers)
```

DNA modifications



Future work

- Eventually support Safari browser (currently only Chrome/Firefox)
- Maybe…
  - Visualize 3D structures
    - file format allows for non-parallel helices, but currently they are displayed in parallel
  - Integrate finite-element modeling for “rough approximation” of CanDo-style 3D structural prediction
    - **codenano** is a related project that has already implemented a web-based API for this (https://dna.hamilton.ie/2019-07-18-codenano.html)
  - Collaborative editing (a la Google Docs)
- Why *maybe*?
  - Primary goal of scadnano that I don’t want to compromise: *simple* and *easy to use*.
  - So I have a secondary goal of being *unambitious* with it. (hopefully keeping it broadly useful)

Try it out!

- scadnano.org
- help available from menu:
  - web interface help
  - Python scripting library help
  - web interface tutorial for making 24-helix DNA origami rectangle
  - Python scripting tutorial for making 24-helix DNA origami rectangle
  - Python scripting API
- feature requests/bug reports at
  - https://github.com/UC-Davis-molecular-computing/scadnano/issues
  - https://github.com/UC-Davis-molecular-computing/scadnano-python-package/issues

Thank you!