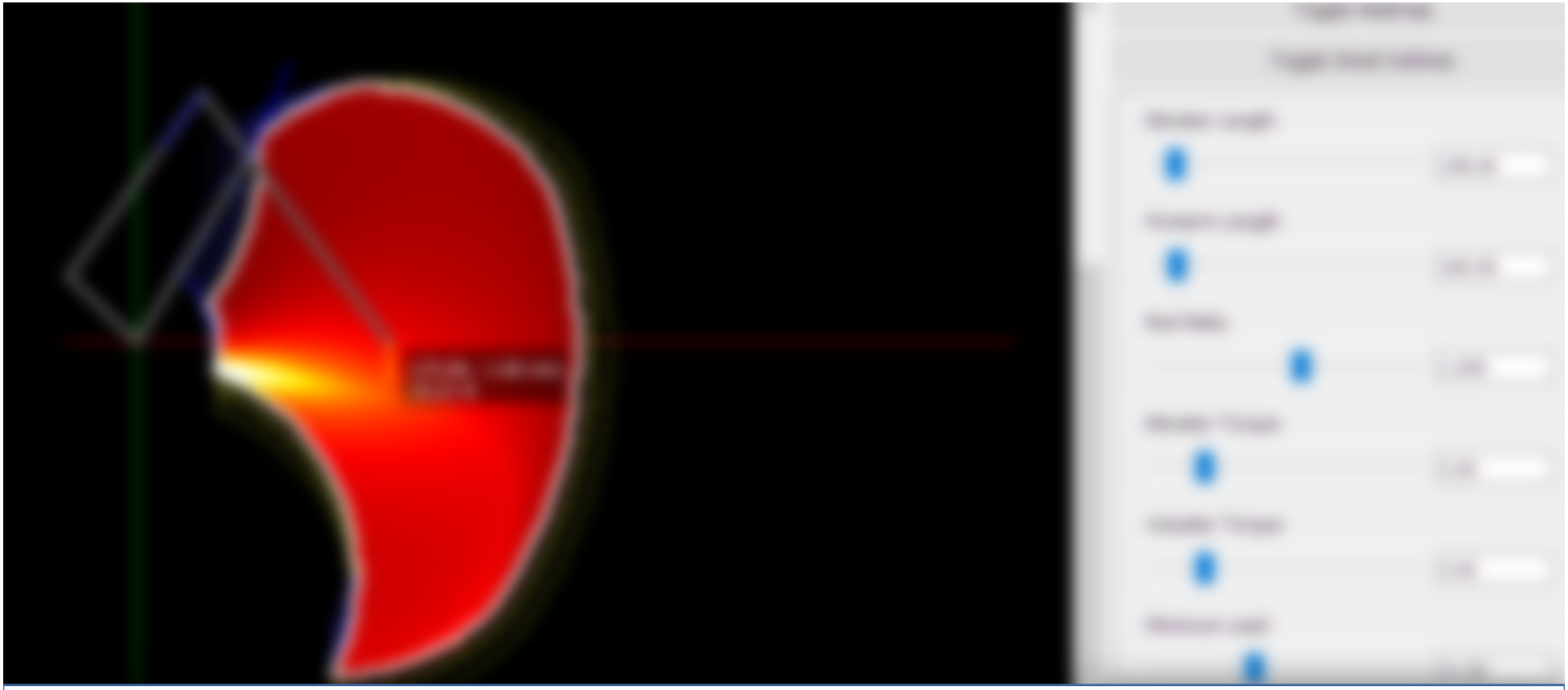
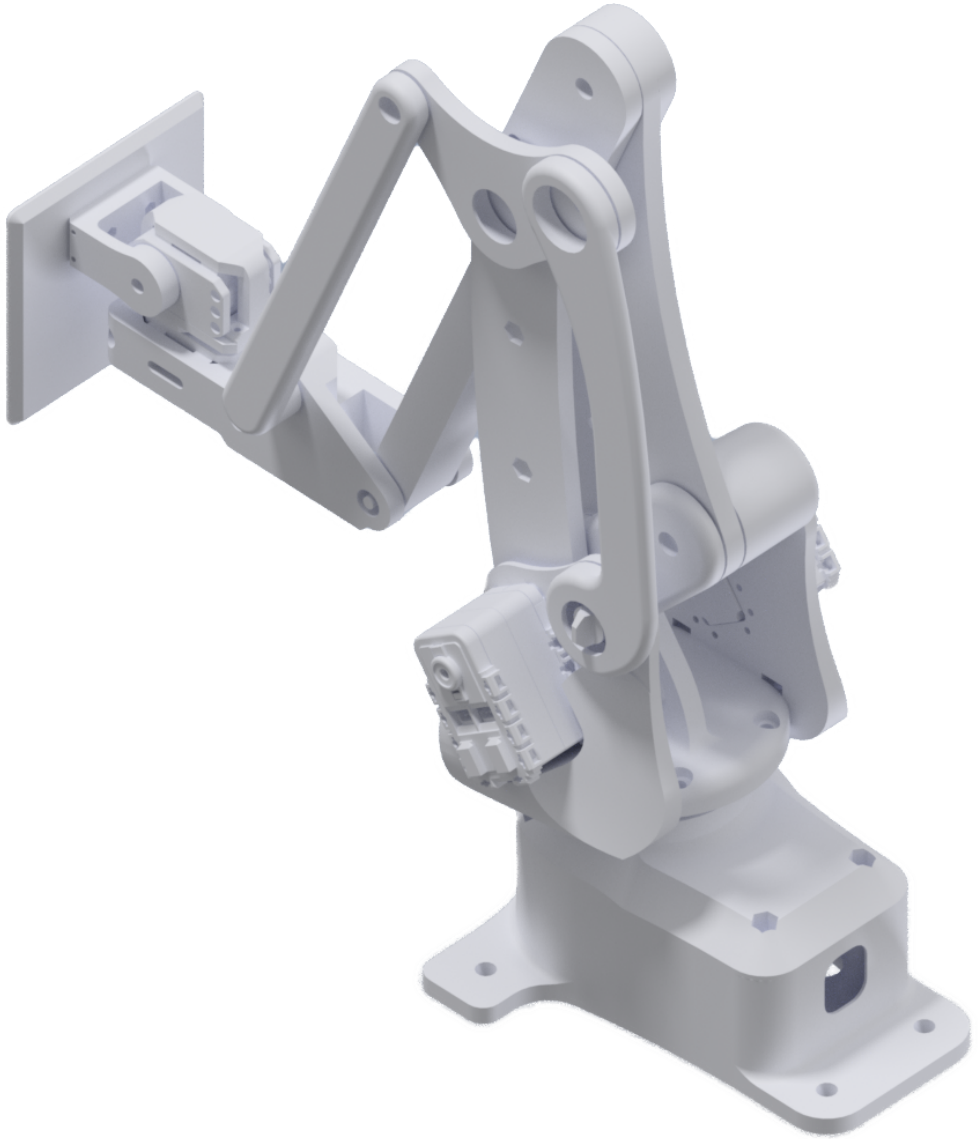


RoboVis

Alistair Wick

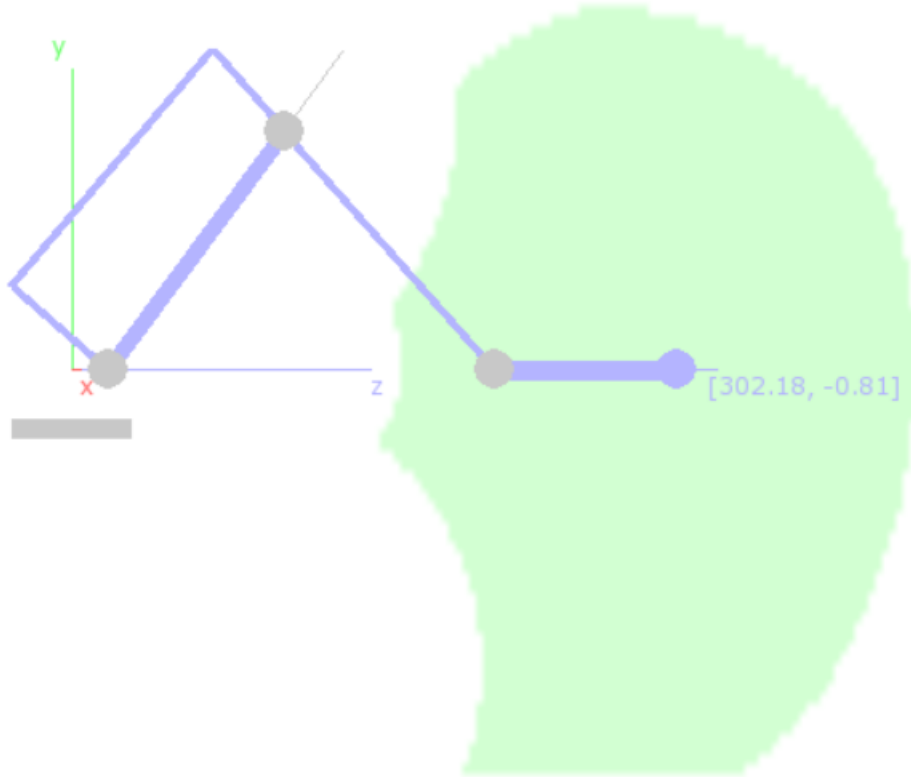
Background

- Designed a small robot arm
- 3 degrees of freedom
- 3D printable
- Controlled with a Python App



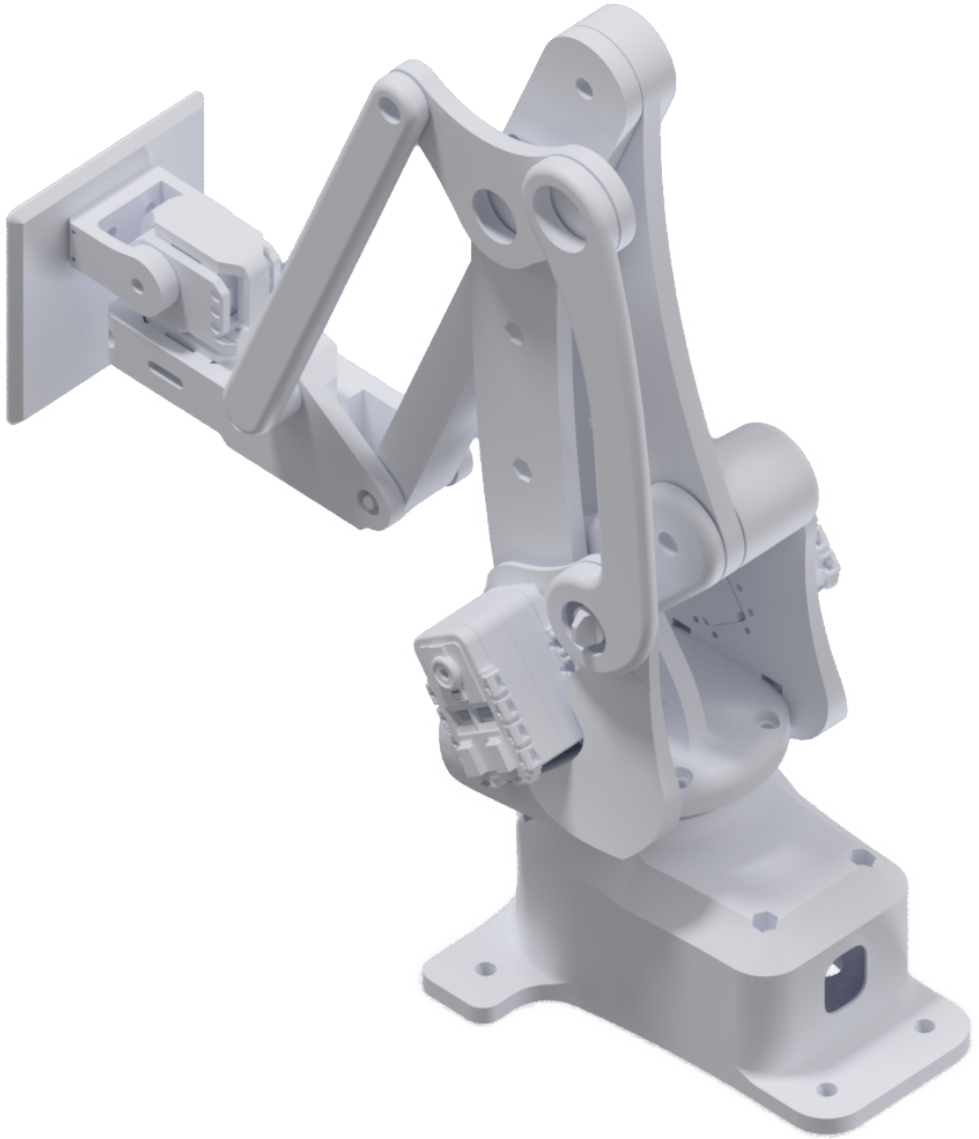
Background III

- Old control app shows where the arm can reach (workspace)
- Uses inverse kinematics based on the arm's dimensions and other parameters



Background II

- The fact it's 3D printed is interesting
- 3D printing is great for small runs of custom objects
- Why have a static design?
- Idea: Let users design their own arm for printing



The Idea

- Make a tool to help users design robot arms!
- Should guide towards useful solutions
- Tool produces ‘config’ files describing dimensions, motor power, constraints, etc.
  - Config can be fed into an OpenSCAD script (or similar program)
  - … generates 3D files for printing!
- Assume users have an idea of the load and reach requirements

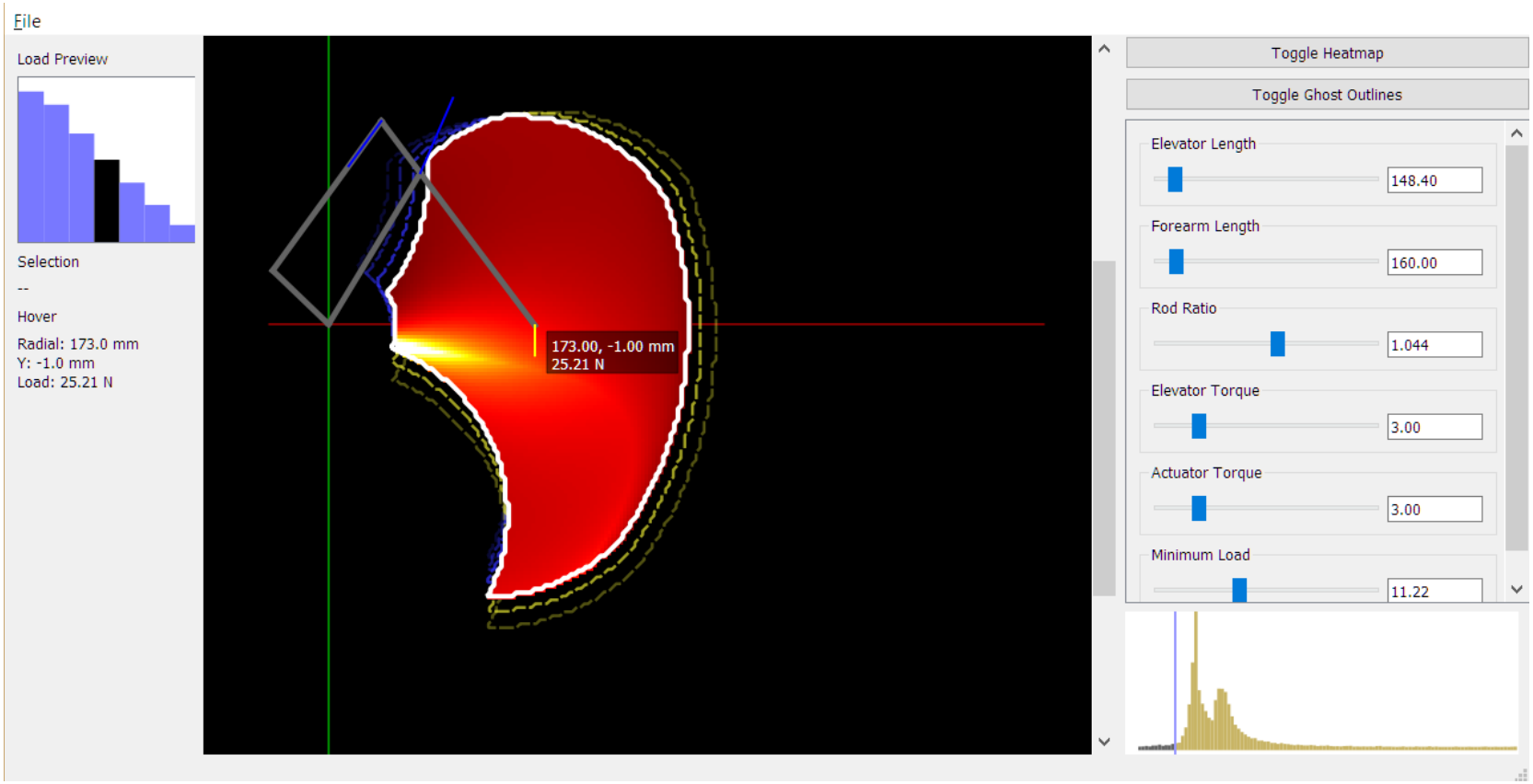
Why not use the existing app?

- Painfully slow
  - Can’t change config in-app: alter file and restart app
  - 10+ seconds to calculate workspace for *one* arm
- Built as a middle-man control app, not for design
  - No advanced GUI
  - No load calculations

Objectives

- Essentially a configuration space exploration tool
- Display workspace and load capabilities
- Tool which will guide towards a good solution
  - User has some idea of what they need
  - Help them find the configuration that achieves that
- Responsive – fast iteration
- Ability to output config files

RoboVis - Overview

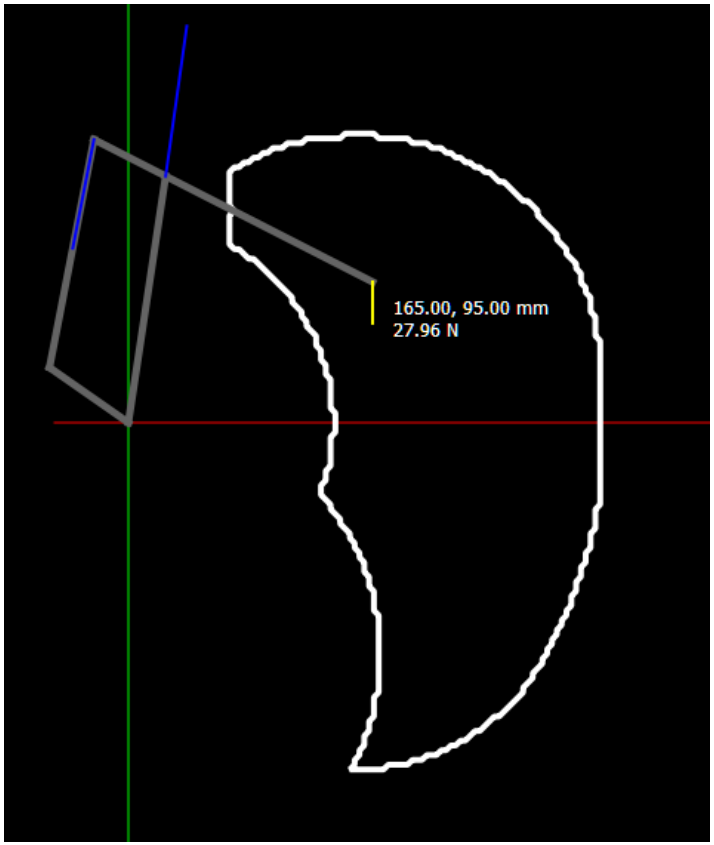


Usage

- Application has a “current” configuration, and a particular parameter of interest
- The main view visualizes the results for this configuration
  - Expected reach
  - Expected load capabilities
- Current config adjusted using the sliders/value boxes on the right hand side
- Config can also be saved to/loaded from a YAML file

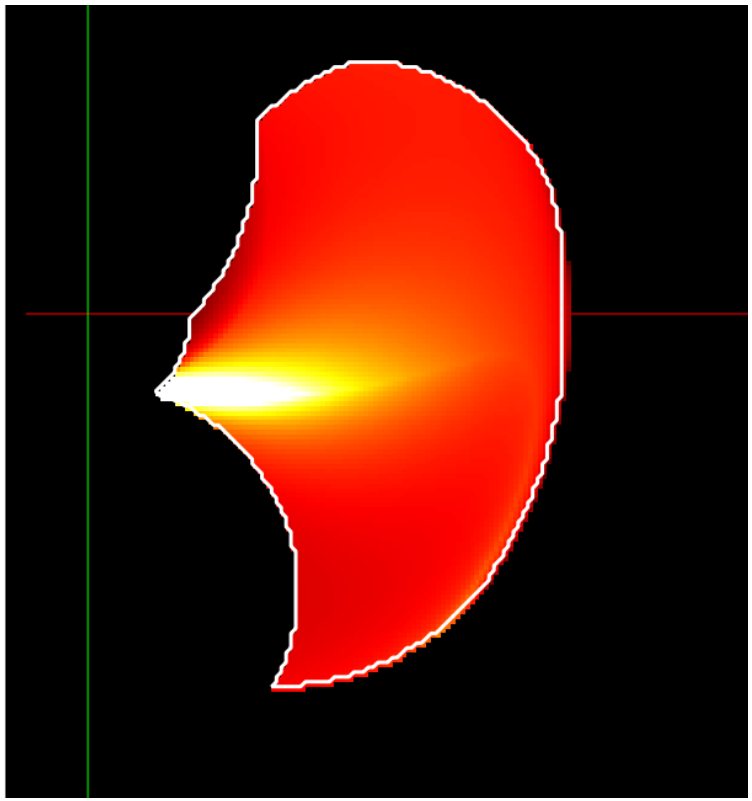
Workspace - Outline

- The space the end-effector can reach
- IK run across sample grid
- Contour-map around points with a valid solution

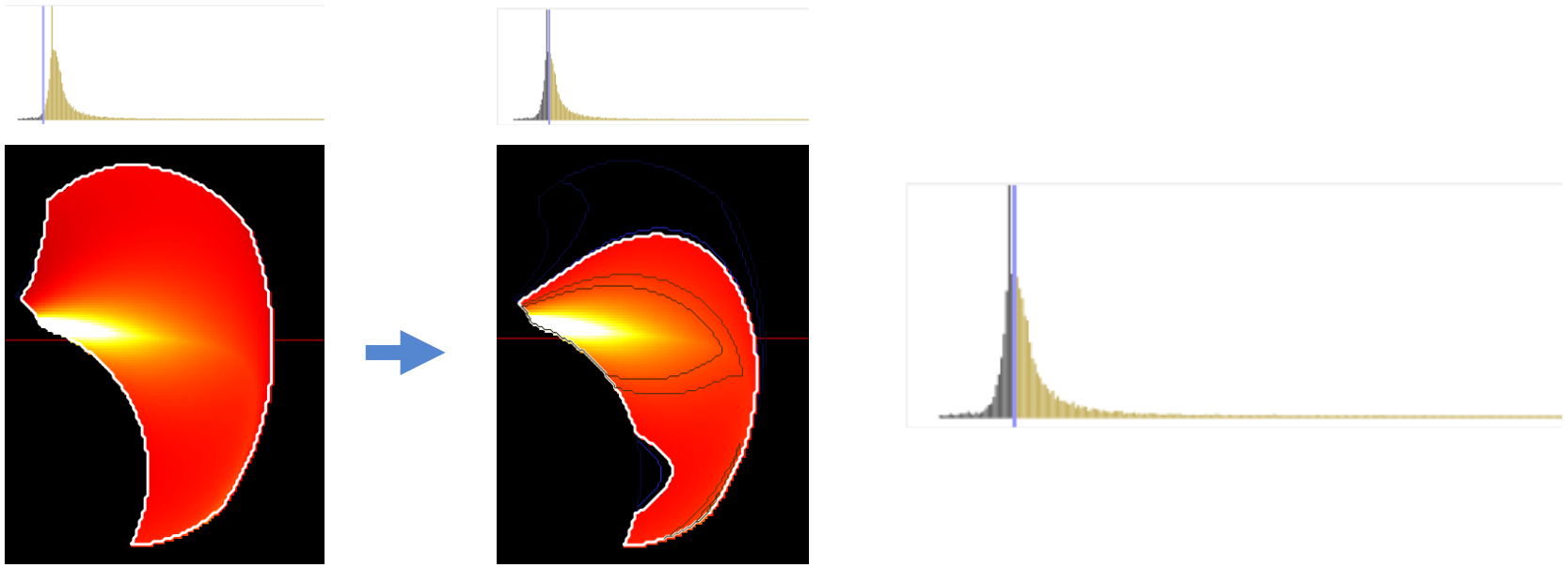


Load - Heatmap

- Shows the distribution of maximum load for the current configuration
- Max load calculated for static case with specified motor torques



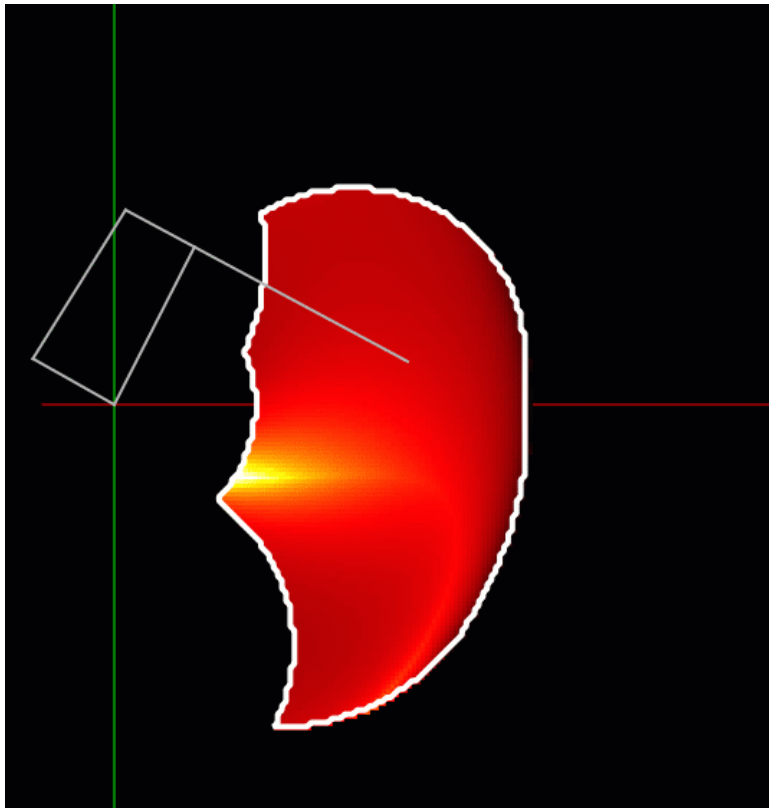
Load Histogram/Slider



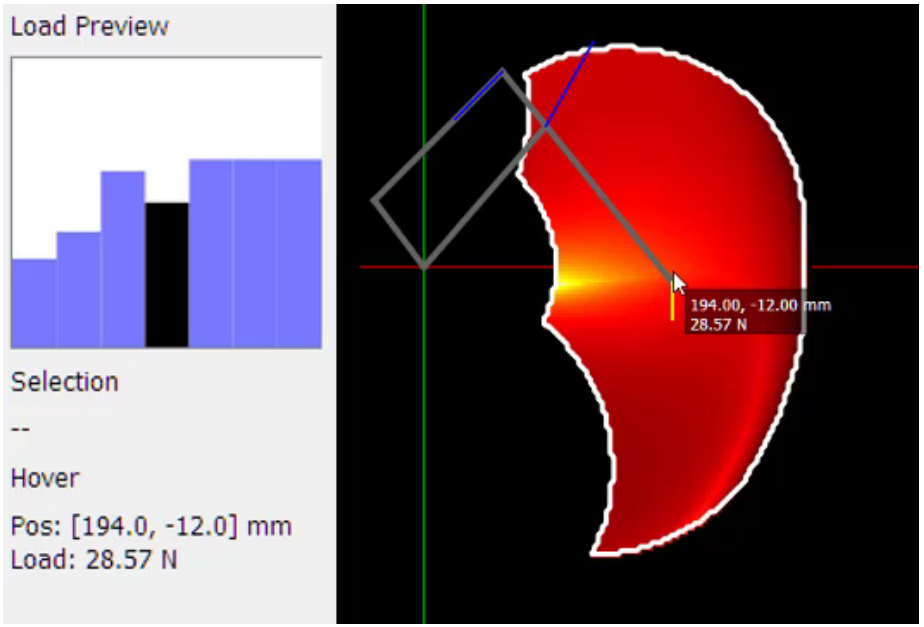
- Minimum load slider
- Coupled with histogram of load

Basic Interaction

- Shape and heatmap respond as the user alters parameters
- Visualization of the arm can be shown for a selected point
- Arm updates its dimensions and pose

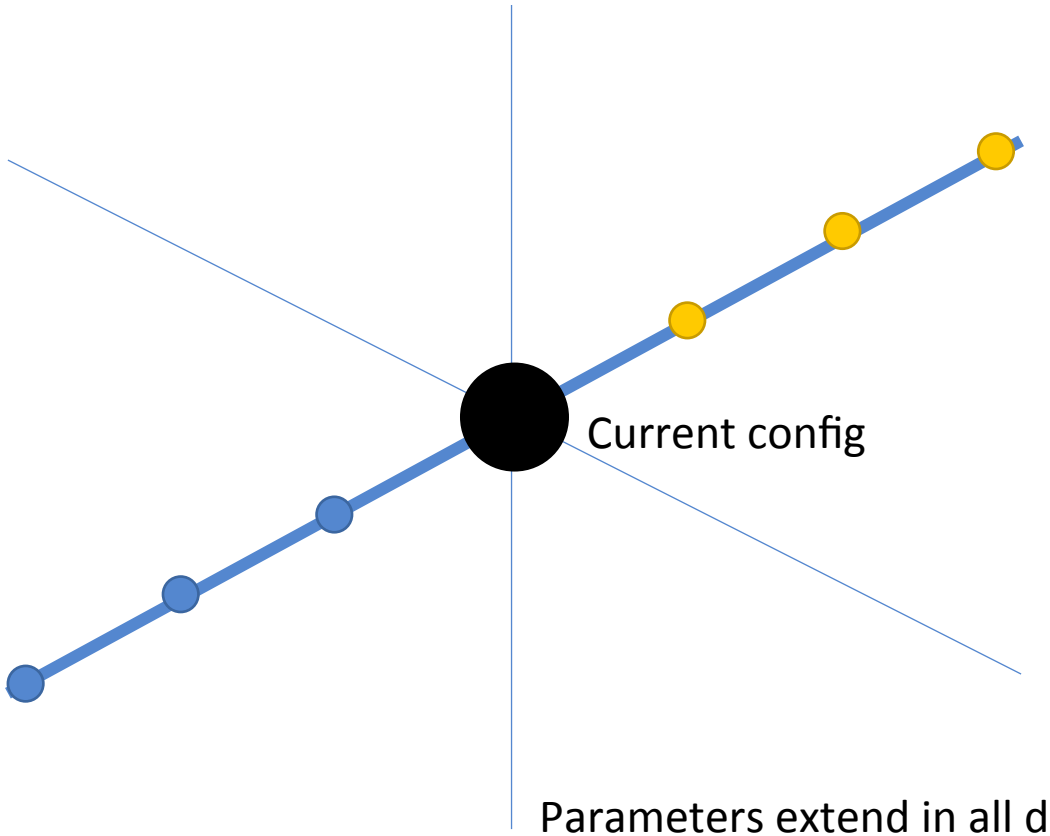


Inspection



- Hover over the vis to see coordinates in mm + max load in newtons
- Click to select inspection point
- Bar chart shows how load at a point will change with the current parameter

Parameter space

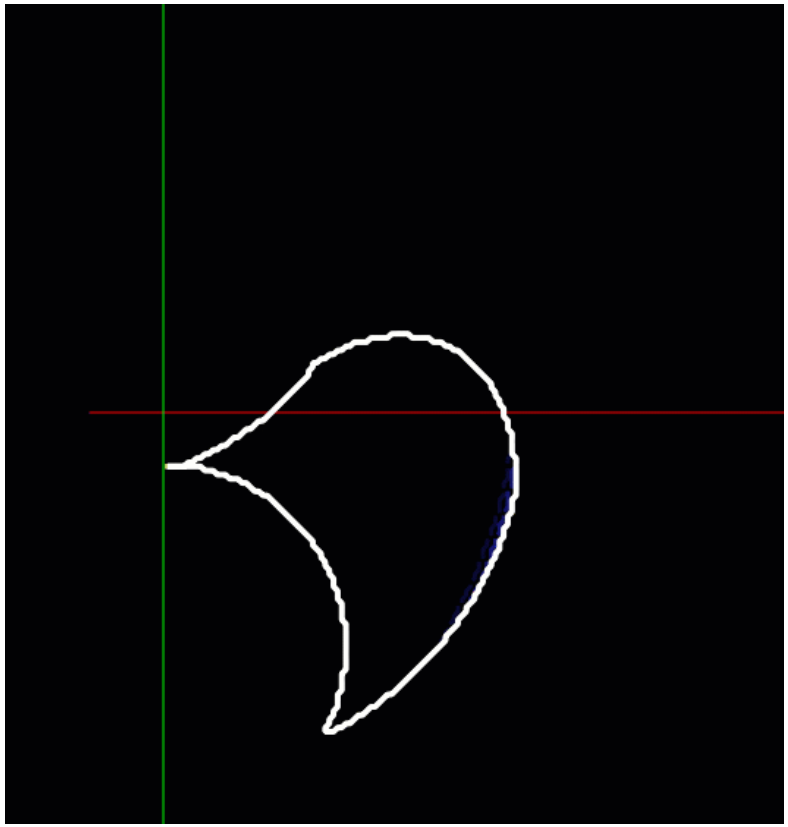


How will the results change with a particular parameter?

Parameters extend in all directions

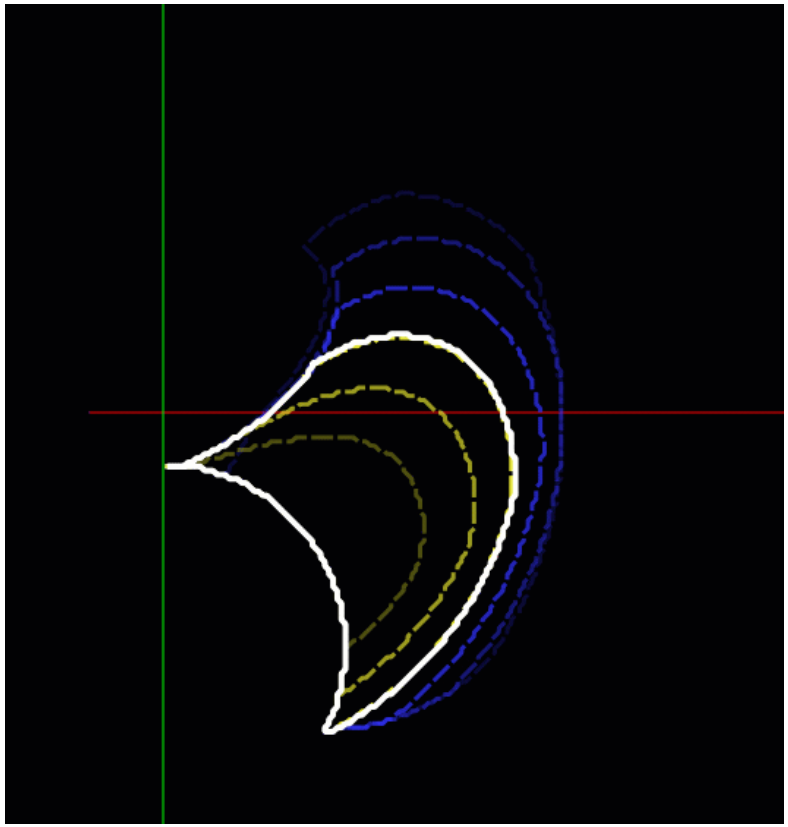
Ghosts – Preview Outlines

- Extra outlines around the main one
- Show how the reachable area will change for different parameters
- Blue for -ve change
- Yellow for +ve change



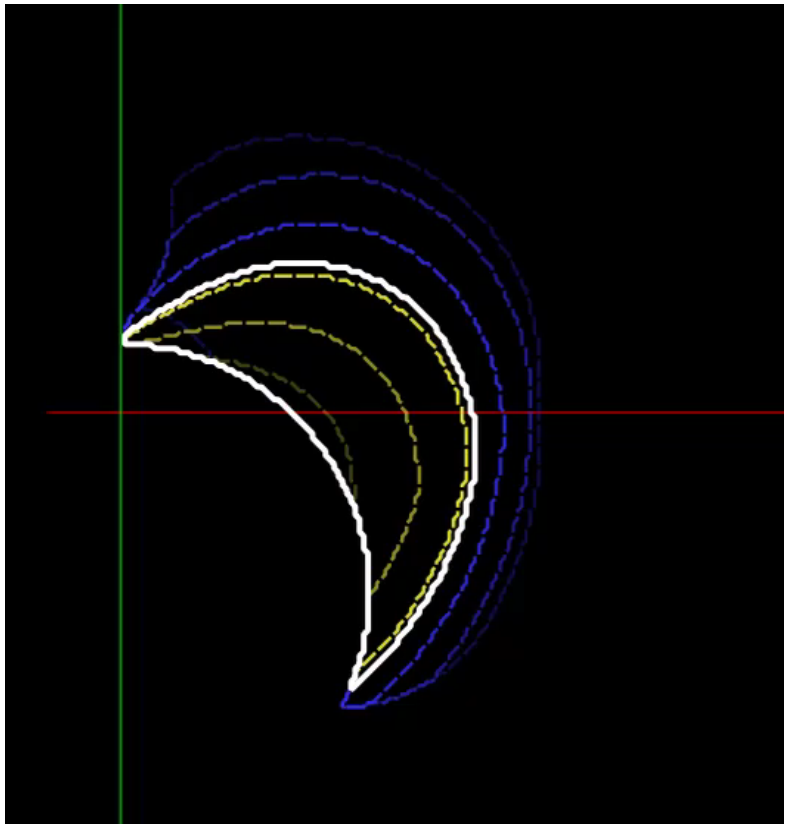
Ghosts – Preview Outlines

- User can then ‘slide through’ the configurations in real time
- Ghosts are updated (recalculated) on the fly
- Opacity shows distance from current config



Ghosts – Preview Outlines

- Switches between parameters are instant!
- Ghosts for the other parameters are kept updated



Technical

- Lots of work making this responsive!
- NumPy is awesome
  - Reformulated IK code into big matrix operations
  - Improved IK solving performance by around 200x over naïve Python implementation
- Ghost outlines are an ‘embarrassingly parallel’ problem
  - Calculations offloaded to pool of worker processes
  - Spreads work across all CPU cores
  - Results are presented as they become available

Thanks!

Questions?

Asynchronous Calculations