Fabrication *a Lavoisier*

An Interactive data driven design system for designing functional objects that can be directly manufactured.
Goals

• Easy to be used by casual users
• Guarantee that object is fabricable and viable
• Instant feedback and suggestions
System Design

- Design UI
- Template Representation
- Evaluation Engine
Design UI
Design UI

• Allow template visualization
• Suggestions of relevant components
• Operations: replacement, addition, removal
• Adjusts user specifications to guarantee structural feasibility - or gives a warning/suggestions
System Design

- Design UI
- Template Representation
- Evaluation Engine
Template Representation

• Hierarchy of Templates

  - table
  - top
  - legs
  - leg1
  - leg2

• Templates:
  • DOF (q)
  • constraints on q
Template Representation

• Hierarchy of Templates

  - table
    - top
    - legs
      - leg1
      - leg2

• Templates:
  - DOF (q)
  - constraints on q

• Elements:
  - fabrication rule
  - geometry
  - mapping function
  - patches
Constraints

- Every node (template) has a list of constraints

- Table:
  - Top and legs touch

- Legs:
  - Symmetry between the legs
  - Ranges of legs
Constraints

• Every node (template) has a list of constraints
• Constraints are represented as a graph
Constraints

• Every node (template) has a list of constraints
• Constraints are represented as a graph
System Design

- Design UI
- Template Representation
- Evaluation Engine
Evaluation Engine

- Operations
  - replacements
  - insertions
  - removal

✅ constraints and connectors!
Steps

• use data driven or grammar based rules to match user operations as best as possible
• stability analysis to validate
Replacements

+ = or

- Two lists of constraints/connectors that need to be merged
- metric for matches
- dealing with redundancies/absence
Additions

\[
\pi + \mathbb{I} = \mathbb{I}
\]

- Only one lists of constraints/connectors
  - evaluating if constraints can be translated
  - dealing with absence
Example
Example
Current Work

• Improving the UI
• Allowing suggestions
• Coming up with more robust matching rules for constraints and components
• Combining with the simulation packages
• Adding more semantic information to the database (?)