Automatic Design of Task-specific Robotic Arms

Ruta Desai, Margarita Safonova, Katharina Muelling and Stelian Coros

NREC
National Robotics Engineering Center

Carnegie Mellon University
The Robotics Institute
Custom, task-specific robotic arms instead of off-the-shelf arms

KUKA

UR

ABB
Custom, task-specific robotic arms instead of off-the-shelf arms

Hard to adapt standard robot arms when task requirements change
Custom, task-specific robotic arms instead of off-the-shelf arms.

Image courtesy: Modbots, HEBI Robotics
Tools for on-demand design of custom robot arms
Current robot design tools are tedious and complex

Design and configure

Simulate and test
All-in-one tool for easy visual design and simulation

Desai et al., Computational abstractions for interactive design of robotic devices (ICRA 2017)
All-in-one tool for easy visual design and simulation

Virtual pin connectors define possible connections of parts

Highlight possible connections with current design

Desai et al., Computational abstractions for interactive design of robotic devices (ICRA 2017)
Given a library of modular building parts and a user-supplied task, our design synthesis method aims to generate the corresponding valid robotic arm design. This process involves using connection rules to combine parts from the library into a valid robot design.

The recursive approach for synthesizing new designs is as follows: for node $D_1$ with children $D_2, D_3, \ldots$, the child with the lowest $g(h)$ is chosen to traverse. Instead of brute force building of the design space, we leverage an existing informed search method, such as A* search, to efficiently explore the design space and find the simplest valid robot design.

Automatically generated designs are then mapped to a space of tasks the robot can perform, such as sealing, welding, or other operations. This allows users to specify their task requirements and see how different parts can be combined to achieve those tasks in the real world.

Manual editing is enabled through intuitive drag-and-drop tools, allowing for further customization of the designs. The interface provides an environment with obstacle detection and obstacle avoidance capabilities, enabling users to specify target paths for the robot arm to follow, as well as designing an environment with obstacles to navigate around.

The recursive approach for synthesizing new designs is as follows: for node $D_1$ with children $D_2, D_3, \ldots$, the child with the lowest $g(h)$ is chosen to traverse. Instead of brute force building of the design space, we leverage an existing informed search method, such as A* search, to efficiently explore the design space and find the simplest valid robot design.

Automatically generated designs are then mapped to a space of tasks the robot can perform, such as sealing, welding, or other operations. This allows users to specify their task requirements and see how different parts can be combined to achieve those tasks in the real world.

Manual editing is enabled through intuitive drag-and-drop tools, allowing for further customization of the designs. The interface provides an environment with obstacle detection and obstacle avoidance capabilities, enabling users to specify target paths for the robot arm to follow, as well as designing an environment with obstacles to navigate around.

The recursive approach for synthesizing new designs is as follows: for node $D_1$ with children $D_2, D_3, \ldots$, the child with the lowest $g(h)$ is chosen to traverse. Instead of brute force building of the design space, we leverage an existing informed search method, such as A* search, to efficiently explore the design space and find the simplest valid robot design.

Automatically generated designs are then mapped to a space of tasks the robot can perform, such as sealing, welding, or other operations. This allows users to specify their task requirements and see how different parts can be combined to achieve those tasks in the real world.

Manual editing is enabled through intuitive drag-and-drop tools, allowing for further customization of the designs. The interface provides an environment with obstacle detection and obstacle avoidance capabilities, enabling users to specify target paths for the robot arm to follow, as well as designing an environment with obstacles to navigate around.
Automatic inverse design

Part library

Desired robot arm's end-effector motion

Synthesized design
Search-based automatic design

Nodes are collection of components/ partial designs
Edges are connections between components
Search-based automatic design

\[ f(N) = g(N) + h(N) \]

Nodes are collection of components/partial designs
Edges are connections between components
Search-based automatic design

**A* search**

\[ f(N) = g(N) + h(N) \]

heuristics: design’s functionality for the task

\[ h(N) = \mathcal{E}_{IK} \]

path cost: design’s simplicity

\[ g(N) = \delta(N) \]

Nodes are collection of components/ partial designs

Edges are connections between components
Automatic design in action
Preliminary results

Given a path and robot base:

<table>
<thead>
<tr>
<th>2 DOF</th>
<th>3 DOF</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Solution time:
- 2 DOF: ~ 210 sec
- 3 DOF: ~ 780 sec
Preliminary results: validation experiments

Fig. 4. (a) To validate our approach, we synthesize arm designs to follow trajectories that correspond to manually created designs (denoted as original designs). Our approach not only generates valid designs in all cases, but also finds simpler designs (with lower DOF) to follow trajectories that were originally generated using robotic arms with higher DOF in some cases, as shown in (b). Actuators and end-effector are highlighted in blue.

approach by synthesizing robotic arms that execute user-specified motions. We also presented an interactive design environment that allowed users to define their task requirements and to intuitively interact with the automatic design synthesis. Our preliminary results show the ability of our system in synthesizing valid, and as simple as possible arm designs. Currently, the target motion trajectories for robot are user-specified. To enable design in much complex scenarios with large number of obstacles, we plan to integrate a motion planner in our system. We will also test our system to synthesize designs for multiple target trajectories in the future. Finally, we will apply our automatic design approach for synthesizing other robotic systems, to test its generality.

REFERENCES

Preliminary results: validation experiments

Fig. 4. (a) To validate our approach, we synthesize arm designs to follow trajectories that correspond to manually created designs (denoted as original designs). Our approach not only generates valid designs in all cases, but also finds simpler designs (with lower DOF) to follow trajectories that were originally generated using robotic arms with higher DOF in some cases, as shown in (b). Actuators and end-effector are highlighted in blue.

REFERENCES


Limitations and next steps

Encoding other desirable properties of design within the search

Original (4-DOF)  Synthesized (4-DOF)

light wrist  heavy wrist

References


Limitations and next steps

Encoding other desirable properties of design within the search

Integrating motion planner for increasing accessibility of design process
Limitations and next steps

Encoding other desirable properties of design within the search

Integrating motion planner for increasing accessibility of design process

Experiments with more complex task scenarios and testing on hardware
Questions?

robot designs consisting of different collections of parts also enables users to define their task requirements easily. To support user design of custom task-specific robots with their designs using the physics-based simulation in the task environment. Various end-effectors such as a sealing gun or a welding machine, according to their task requirements. Based on these specifications, our algorithm can be constructed even with a small-sized part library. To keep adding more components, brute force construction is not well-suited for a user. The recursive approach for synthesizing new designs is motivated by the following observation. Consider a robot design that can execute a user-specified motion, one of its children designs might be. This is represented by a node in the tree that is most promising.

Automatic Design Using Informed Tree Search

A* search can be well-represented with an acyclic tree of all possible designs. Specifically, A* chooses nodes that minimize the cost function, as defined by the connection rules. As a node is traversed, the cost function is updated to valid designs, capable of executing user-specified motions. One can create new designs with increasing number of components using connection rules, for creating a library of modular parts, we develop an intuitive graphical interface before assembling a real design. Finally, users can test all possible designs.

Manual editing is enabled using intuitive drag-and-drop operations. The edge between a parent node and its children represents the current cost of the design. The recursive approach for synthesizing new designs is as follows. The node that is most promising is added to the tree. This process is repeated until a valid design is found (see fig. 2(b)).

Questions?

rutad@cmu.edu