The Robot Engineer

Claude Sammut[1], Raymond Sheh[2], Adam Haber[3], and Handy Wicaksono[1]

[1] School of Computer Science and Engineering,
University of New South Wales, Sydney, Australia
[2] Department of Computing, Curtin University, Perth, Australia
[3] Broad Institute, Cambridge, MA, USA

Abstract. We described preliminary work on designing a "robot engineer". Somewhat similar to the idea of a robot scientist, the robot engineer is a closed-loop system that designs tools for a robot, and even complete robots, that are tested in simulation, manufactured and realised using 3D printing. The artefact is then evaluated on real-world tasks, feeding back to the original design. The system builds on ILP techniques originally developed for learning tool use by a robot. These are extended with methods for transforming the functional specification produced by the ILP system into a design that is suitable for manufacture.

1 Introduction

With the advent of affordable 3D printing, it is possible to construct custom made tools on-demand and, with the addition of low-cost computers, sensors and actuators, it is even possible to construct custom made robots. This ability is particularly useful when an autonomous system is required to operate in a new environment or to perform a new task. For example, in urban search and rescue, it is often difficult to anticipate how access can be gained to confined spaces and what kind of configuration is needed for the robot to complete its task. Similarly, in agile manufacturing, retooling automation equipment for new tasks consumes considerable resources. Reducing this cost allows manufacturers to respond more readily and efficiently to changes in the market. In this paper, we describe methods for automatically designing and constructing 3D printable tools that can be used by a robot in response to novel and changing environments and tasks. The design is derived from specifications learned by an ILP system.

Response robots are being used increasingly in emergencies [5]. At present they are mainly sent into a disaster site to perform preliminary surveillance before human rescuers enter a dangerous environment. Often, it is impossible to know, in advance, what capabilities will be required by the robot to complete its task. For example, gaining access to and working in disaster sites, such as a collapsed building, is problematic because they contain unexpected obstacles, damaged infrastructure, narrow spaces, etc. Thus, it is difficult to anticipate how a robot should be configured.

With 3D printing, it is possible to create custom tools suited to a unique situation at the disaster site, and even manufacture a complete robot [6]. Instead

of travelling with a collection of robots, a rescue team might travel with a small number of robots, a set of standard parts, and high speed 3D printing equipment. Once the situation has been determined, robots and tools suited for the job can be rapidly produced and deployed. In this work, we address the problem of automatically designing robots and their tools, since design expertise is not a skill that human rescuers or manufacturing technicians usually possess.

We build on previous work by Brown and Sammut [1] and Haber [4], who developed ILP methods for a robot to learn to use a tool by observing another agent using that tool. This is extended so that instead of learning how to use an existing tool, the system will propose a new tool to suit the task, if none is available. The same technique can also propose complete robots by adapting designs from a library.

There are several stages in the creation of a design for a tool or robot. The first is to determine a functional specification for the tool, in terms of its type and shape. The second is to work out a design for the tool, based on this specification, and determine how it can be manufactured to satisfy this specification. Given the functional specification, it is necessary to produce a physical design for the tool, incorporating considerations material strength, weight, etc. In addition, because fused filament 3D printing works by depositing one layer of material on top of another, some objects cannot be printed directly. For example, if there is an overhang, where there is no supporting material upon which to deposit the next layer, it is not possible to make this object without some addition work. In this case, the design must include a temporary support structure that can be removed after the entire object has been printed. The order in which the tool is printed, and the path that the printer takes, also affects the strength and stability of the tool.

The remainder of this paper addresses the two major tasks in tool specification and manufacture: (1) determining a functional specification and (2) converting the functional specification into a design that can be realised in practice.

2 Producing a Functional Specification of a Tool

The method for producing functional specifications for tools extends previous work by Brown and Sammut [1] and Haber [4]. In this setting, a robot learns to use an existing tool by a single observation of tool use by a trainer. Thereafter, the robot performs experiments, choosing different tools and applying them differently to generalise the description of the tool and the ways in which it can be manipulated. For example, the trainer may demonstrate using a hook to pull an object out of a confined space. The robot's experiments include selecting a new tool of a different shape to discover the features of the tool that are required to perform the task of pulling an object. Other experiments vary the positioning of the tool to learn how it must be handled. We extend this method to develop functional specifications for tools that do not yet exist. We assume that there has been prior learning of the kind just described, and the task of the system is to use

this background knowledge to produce functional specifications for new tools. To avoid repetition, when we talk about producing functional specifications for a tool, we include potentially doing so for the robot.

We define a tool by the way in which it enables an action to be performed, where an action is represented by a STRIPS-like formalism [3], extended to include pragmatic information for the subsequent construction process. For example, if a robot is required to open a door but lacks an appropriate end effector, the use of a tool such as a simple hook enables the `open_door` action to be performed. Thus, a *tool action* is defined to be one that changes the properties of one or more objects in a way that enables the preconditions of one or more other actions in the agent's plan library.

If an appropriate tool is not be available, but we know the desired properties, we can produce the functional specification for one. Using the example of pulling a box out of a tube with a hook, we define the following actions:

```
position-tool(Tool, Box)
  PRE  in-gripper(Tool), gripping
  ADD  tool_pose(Tool, Box), obstructing(Tool, Box)
  DEL  -

pull-from-tube(Tool, Box, Tube)
  PRE  tool_pose(Tool, Box), gripping(Tool), in-tube(Box,Tube)
  ADD  -
  DEL  in-tube(Box, Tube)
```

The first action represents the robot getting itself into the correct position so that the tool can be applied. The predicate `tool_pose(Tool, Box)`, which expresses this position, is learned in the experimentation stage. A side-effect of this action is that the tool is obstructing the object. Later, when the robot tries to pick up the object, this side-effect will have to be undone. The `tool_pose(Tool, Box)` effect of the position-tool action becomes a precondition of the tool action, pull-from-tube.

The experimentation phase refines the definition of `tool_pose(Tool, Box)`:

```
tool_pose(Tool, Box) :-
    in-tube-side(Box, Tube, Side),
    attached-side(Tool, Hook, Side),
    touching(Hook, Box, back),
    attached-angle(Tool, Hook, rightangle),
    attached-end(Tool, Hook, back).
```

Note the "attached" predicates. These describe some of the shape features of the tool, which is just a stick with a right-angled hook attached at the end. It must be placed inside the tube, and on the side such that the hook is touching the box to be removed from the tube.

Assuming that the robot has learned to use tools for tasks such as extracting objects from confined space, placing timber to shore up ceilings, opening doors,

etc. Let us now suppose that it is faced with a task for which no tool is available. It is possible that the right tool exists in its library of action models, in which case, the "structural" goals in the `tool_pose` clause above, can used to generate a functional specification for the desired tool. However, if the plan library does not contain an appropriate tool, then further reasoning is required.

Suppose the problem the robot has a new problem: to approach a door, open it and move through to the next room. A planner uses the robots library of action models to construct a sequence of actions to achieve the goal of being in the next room. The problem is that the robot has no end effector that can grip the door handle. Thus, there is a gap in the plan resulting from the door opening action not having some of its preconditions met. However, these preconditions may suggest the properties that a tool should possess so that it can be used to manipulate the door handle. This gives a starting point for determining the functional specifications of the tool. Structural predicates, such as the "attached" predicates in the `tool_pose` clause above, tell us the qualitative properties of the tool.

The parameters of the tools functional specification can be determined experimentally, by trial and error. Fortunately, we do not have to manufacture many actual tools to perform these experiments. Instead most experiments can be done in simulation before a physical tool is designed and built. Sushkov [7] developed a learning system in which a real robots world is modelled in simulation. When faced with a trial and error learning task, it first performs "thought experiments", that is, it runs experiments in the simulator to determine which ones, if performed in the real world, would yield the highest information gain. This method is being adapted for developing functional specifications for tools.

As described above, the planner produces predicates that can be used as qualitative constraints on the design of the tool. Within those constraints, the system must search for the quantitative parameters of the tool. For example, how long should the hook be? What configuration of gripper will apply the correct force to the door handle? This is a constraint solving and optimisation task that can be tested in simulation. Once a plausible solution has been found, this can be handed over to the manufacturing phase for testing in the real world.

3 Tool Manufacture

Manufacturing a tool given a functional specification builds on work by Sheh, Komsuoglu and Jacoff [6] in constructing robots using fused filament 3D printers. The robot must take the functional specification that results from the planning phase, above, and produce a working physical component using a fused filament 3D printer. In addition to the functional specification, the *tool designer* program must consider constraints relating to integrity, strength, durability, ability to be printed and ability to be mounted effectively on the robot. This physical design must then be turned into a set of instructions to the 3D printer to produce the actual part by a program called the *slicer*, the equivalent of a the machinist in a workshop.

3.1 Tool Designer

Like Brown [1], the STRIPS action model is extended to include pragmatic information about what objects are affected by the action and how they are affected. This information is expressed as semi-numerical constraints, for example giving the maximum length allowable for a component of a tool. Testing within a simulator is used to the final design if the tool, with parameters that satisfy these constraints. The simulation involves stochastically sampling the possible variations in the manufacture of the tool, positioning of the robot and other objects.

The designer may be unable to find a design that satisfies all of the functional specifications. For example, it may find that a hook of sufficient length to satisfy the functional specification cannot be made strong enough to be practical. This failure is referred back to the planner to determine if the functional specification can be modified. Additional constraints, e.g. on the length, are added to ensure that a new design does not fail in the same way.

3.2 The Slicer

The *slicer* must ensure that the tool design is printable in a way that satisfies the functional specifications and makes use of them to optimise its production.

In this project, we only consider fused filament 3D printers, which produce parts by extruding hot plastic from a moving nozzle onto a platform and building up an object, layer by layer. The slicer must ensure that the tool design is printable in this way. Slicing involves turning an enclosed 3D object into a path for the nozzle to produce that object. The strength of the object depends on the orientation in which the object will be printed and the density with which different parts are printed.

Existing slicers [2] only have access to the geometric shape of the object to be printed. However, having the functional specification can can avoid potential problems. For example, a hooked object may need to be printed in one orientation to achieve the required level of strength, yet to be printed with minimal overhangs it may need to be printed in another orientation. Thus, only a part of the specification can be met. This new constraint can be referred back to the designer so that it can adjust the design and amend its rules for future designs. For example, a new design may avoid this problem while satisfying the functional specification by rotating part of the tool by 90° and compensating for this rotation in the controller.

4 Evaluation

The US National Institute of Standards and Technology (NIST) has defined standard test methods for response robots. These include inspection tasks, such as looking in confined spaces to determine the state of a victim; manipulation tasks, such as delivering aid to a trapped victim, opening doors; and locomotion,

including traversing different types of terrain. The test methods are also used in the RoboCup Rescue Robot league. We use the same apparatus from these test methods to evaluate the designs produced by the system. To obtain statistically meaningful results, simulations of the test methods are used to perform large numbers of repeated experiments. However, simulation alone is insufficient since the physical world cannot be modelled perfectly. Therefore, experiments must be conducted using real printers, robots and test environments, to ensure that the simulation results are meaningful.

5 Conclusion

This paper has described preliminary work on designing a "robot engineer", which is capable for designing a new tool or robot that can be used to achieve a goal given to a planner. The planner uses STRIPS-like action models that have been learned from demonstrations of similar tool use. If no tool exists to complete a task, a new tool is specified by varying previous designs and testing them in simulation. The specification is then based to a "designer" that must convert the functional specification into an artefact that can be realised using 3D printing technology. This must take into account the physical constraints of printer. Sometimes, these constraints make the specification impossible to manufacture. In this case the constraints are added to the tool's action model and a new specification must be produced. The new constraints force the specification phase to eliminate unimplementable designs.

This work is still in its early stages with preliminary work being done on extending the previous tool use learning techniques to produce functional specifications.

References

1. Brown, S., Sammut, C.: A Relational Approach to Tool-use Learning in Robots. In: Inductive Logic Programming, Springer Berlin Heidelberg, pp. 1–15. (2013)
2. Evans, B.: Practical 3D printers: The science and art of 3D printing. Apress (2012).
3. Fikes, R., Nilsson, N.: STRIPS: a new approach to the application of theorem proving to problem solving. Artificial Intelligence 2, 189–208. (2012)
4. Haber, A.: A system architecture for learning robots. PhD Thesis. The University of New South Wales (2015)
5. Sheh, R., Collidge, B., Lazarescu, M., Komsuoglu, H., Jacoff, A.: The Response Robotics Summer School 2013: Bringing Responders and Researchers Together to Advance Response Robotics. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1862 – 1867 (2014)
6. Sheh, R. K.,Komsuoglu, H., Jacoff, A.: The Open Academic Robot Kit: Lowering the Barrier of Entry for Research into Response Robotics. In: IEEE International Symposium on Safety, Security and Rescue Robotics. IEEE Press (2014)
7. Sushkov, O. O., Sammut, C.: Active Robot Learning of Object Properties. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. E. Guglielmelli (2012)