

# Autonomous Tool Construction Using Part Shape and Attachment Prediction

Lakshmi Nair, Nithin Shrivatsav, Zackory Erickson and Sonia Chernova  
 Georgia Institute of Technology  
 {lnair3, nithinshri, zackory, chernova}@gatech.edu

**Abstract**—This work explores the problem of robot tool construction - creating tools from parts available in the environment. We advance the state-of-the-art in robotic tool construction by introducing an approach that enables the robot to construct a wider range of tools with greater computational efficiency. Specifically, given an action that the robot wishes to accomplish and a set of building parts available to the robot, our approach reasons about the shape of the parts and potential ways of attaching them, generating a ranking of part combinations that the robot then uses to construct and test the target tool. We validate our approach on the construction of five tools using a physical 7-DOF robot arm.

## I. INTRODUCTION

Tools are defined as objects that extend the physical influence of an agent [15]. Among biological sciences, tool creation has been extensively studied in communities of prehistoric humans [26, 25], mammals [7, 28], and birds [6, 15] in order to explore the emergence of more sophisticated levels of intelligence. In robotics, several recent works have explored theoretical frameworks for tool creation [21, 8], 3D tool modeling representations [30, 3, 1], as well as techniques for physical construction of tools and simple machines [10, 24, 16, 19].

In this work, we advance the state-of-the-art in robotic tool construction by introducing an approach that enables the robot to construct a wider range of tools with greater computational efficiency. Specifically, given an action that the robot wishes to accomplish (e.g., scoop) and a set of building parts available to the robot, our approach reasons about the shape of the parts and potential ways of attaching them, generating a ranking of part combinations that the robot then uses to construct and test the target tool (See Fig. 1).

Our work makes the following contributions:

1. We introduce two previously unexplored techniques for attaching objects for tool creation: *pierce attachment* (piercing one object with another, e.g., foam pierced with a screwdriver) and *grasp attachment* (grasping one object with another, e.g., a coin grasped with pliers). For both attachment types we demonstrate fully autonomous techniques for predicting the attachment. Our work on pierce attachments is particularly novel in its use of spectrometer data, and to our knowledge this is the first use of spectral data for reasoning about material pierceability.
2. We experimentally evaluate three previously proposed tool shape representations for their potential use in the context of tool construction, and report their relative strengths.

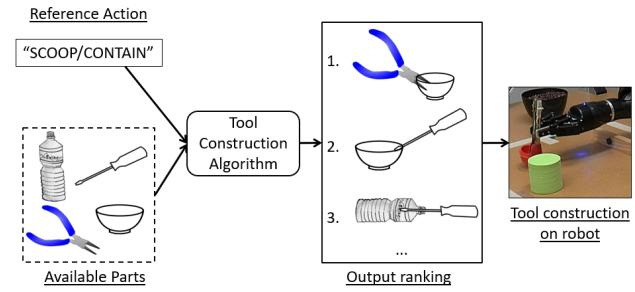


Fig. 1: Tool construction overview: given a reference action and available parts, our approach ranks part combinations that are likely to achieve the reference action. Ranked parts are combined by the robot until a successful tool is constructed.

3. We introduce a novel tool construction reasoning approach that leverages the attachment predictions and shape reasoning techniques described above, in order to effectively rank groups of objects that are likely to result in a tool that accomplishes the required action. The resulting ranking is then used by a physical robot to construct tools.

We validate our approach on a 7-DOF robotic arm. In our experiments, we demonstrate the algorithm's ability to reason about a wide range of objects through autonomous construction of five versions of five different tools.

## II. RELATED WORK

In this section, we summarize prior work on robotic tool-making, 3D tool representations, and spectral sensing.

### A. Robot tool making

In prior work on tool making, Sarathy and Scheutz [21] proposed a theoretical formulation of tool creation in a classical planning framework. Erdogan and Stilman [10], introduced techniques for reasoning about construction of functional structures for navigation. Further work explored the use of environmental objects as simple machines, e.g., levers and bridges [24, 16]. Wicaksono et al. [29], focused on creation of novel tools using 3D printing. More recently, Choi et al. [8] extended the cognitive architecture ICARUS to support the creation and use of tools in abstract planning scenarios. Our work differs from these approaches in that we reason about visual, material properties of objects and present our approach on a physical robot. Our prior work introduced a computational framework for tool construction from environmental objects

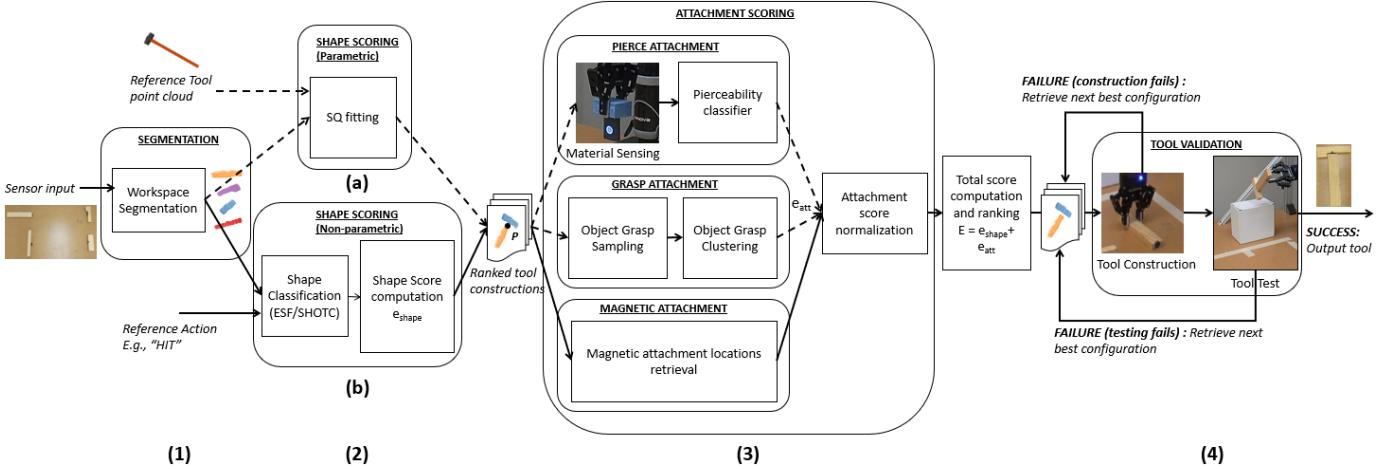


Fig. 2: Overview of our framework highlighting the four key steps involved: segmentation, shape scoring, attachment scoring, and tool validation. Solid lines denote the path taken for the example shown, dashed lines represent alternate paths.

[19], reasoning about one attachment type, namely, magnetic attachments, to output tool constructions that were optimized to match an input example tool. However, the approach required significant parameter tuning and did not scale with number of building parts. In contrast, our work in this paper takes a target *action* as input, and reasons about multiple attachment types to construct a broader range of tools. We further improve on our previous approach, making it more computationally efficient and eliminating the need to fine tune parameters.

#### B. Tool representations

Existing work in robot tool use has introduced several descriptors for representing 3D tool models generated from point cloud data [30, 3, 1]. Descriptors are n-dimensional vectors of real values describing the shape of point clouds. The most widely used of these include non-parametric descriptors, Ensemble of Shape Function (ESF) [30] and Signature of Histogram of Orientations at Centroid (SHOTC) [3, 22]. ESF is a shape descriptor consisting of 10, 64-bin sized histograms describing the shape properties of a point cloud. SHOTC is a descriptor computed at the centroid of the point cloud for a radius equal to its max dimension [22]. Prior work by [22] has explored the use of ESF and SHOTC for affordance learning of objects on a per-part basis using supervised classification. Further, prior work has also explored the use of Superquadrics as a parametric tool representation [1, 19]. Superquadrics (SQs) refer to the family of geometric shapes that includes quadrics, but allows for arbitrary powers instead of just power of two [4].

The relative advantages of the above descriptors are unknown in the context of reasoning about tool construction. As a result, in this work, we compare the performances of ESF, SHOTC and SQ in tool construction.

#### C. Material Recognition

While there has not been prior work exploring material pierceability, the closest approaches are related to mate-

rial recognition. Several vision-based approaches to material recognition have been previously explored [23, 14, 5]. These approaches are sensitive to lighting conditions and viewing angle. Additionally, objects can be deceptive in appearance, resulting in misidentification, e.g., a foam block with wooden texture would be wrongly detected as wood. However, prior work in the use of spectral reasoning for determining material class of objects has shown highly promising results [11] with an overall material class recognition accuracy of 94.6%. Hence, in this work, we use spectral readings obtained from a handheld spectrometer, to reason about materials and determine whether an object can be pierced. To the best of our knowledge, this is the first work to reason about material pierceability from spectral readings with application to tool construction.

### III. OVERVIEW OF TOOL CONSTRUCTION FRAMEWORK

In this section, we introduce our computational framework for tool construction from candidate objects. Fig. 2 presents an overview of our framework, which supports the creation of tools given either a 3D model of a reference tool (similar to [19]) or a reference action. As described below, our contribution focuses on tool construction based on reference actions, and we formulate our tool construction problem as follows:

*“Given a reference action, and a set C of n candidate parts, how can the robot reason about the shape and attachment capabilities of the available parts in order to construct a tool for accomplishing the specified action?”*

We define *attachment points* as locations at which objects can be attached together. In this work, we focus specifically on three different attachment capabilities, the “magnetic” attachment from [19], and the “grasp” and “pierce” attachments novel to this work.

Assuming we are given *n* candidate parts for tool construction, and wish to construct a tool with *m* parts, we are faced with a combinatorial search space of size  ${}^n P_m$ . In this work, we introduce a learning-based framework for tool construction

that is computationally scalable as number of parts increases. Our approach involves four key steps as shown in Fig 2:

1. *Workspace Segmentation*: Our pipeline begins with segmentation and post-processing, which enables the system to identify the candidate parts in the robot’s workspace. We use plane subtraction and Sample Consensus Segmentation (SAC)<sup>1</sup> to identify the candidate parts available to the robot using RGB-D data from a camera mounted over the table. We denote the resulting candidate parts as  $C = \{c_1, c_2, \dots, c_n\}$ .
2. *Shape Scoring*: The shape scoring algorithm, evaluates the shape appropriateness of the candidate parts. As shown in the diagram, there are two classes of approaches for shape scoring: parametric (Fig 2, 2a) and non-parametric (Fig 2, 2b). Parametric shape scoring takes a reference tool as input and uses SQ parameters to find part combinations that optimally match the input reference tool point cloud [19]. In contrast, non-parametric shape scoring uses ESF or SHOTC to find parts that are well suited for performing an input target action. Our contribution focuses on non-parametric shape scoring that takes a reference action as input (section IV-B). Given a list of tuples  $T$  consisting of all possible candidate configurations generated through the permutation  ${}^n P_m$  of candidate parts in  $C$ , both shape scoring methods output a shape score indicating the shape fitness of the parts in each tuple.
3. *Attachment Scoring*: Given the tuple ranking from the previous step, our attachment scoring algorithm (section V), evaluates whether the candidate parts can be attached via pierce, grasp or magnetic attachments. Specifically, the algorithm outputs an attachment score, which, combined with the shape score, is used to generate a final ranking of part combinations indicating the best part configurations for the tool construction.
4. *Tool Validation*: Given the final ranking of parts, the robot constructs the tools by joining the parts specified by the best-rated configuration using the output attachment points and attachment type. The robot then evaluates the constructed tool for its task suitability by applying the reference action on the tool. In this work, we assume that the robot can observe whether the tool succeeded and that the action trajectory is pre-specified. Alternatively, the action trajectory could be learned from demonstration [20], including, if necessary, adapting the original action to fit the dimensions of the new tool [12, 13]. If tool construction fails or the tool fails at performing the reference action, the robot continues to iterate through the ranked list of tuples until a successful tool is found.

In the following sections, we discuss the shape scoring and attachment scoring modules, highlighting our contributions with respect to non-parametric shape scoring and attachment scoring in detail. The notations used in the following sections are shown in Table I.

<sup>1</sup>The implementation was provided by the PCL library

TERM	DESCRIPTION	FUNCTION	DESCRIPTION
$C$	Candidate parts $c_1 \dots c_n$	<i>AttachmentFit()</i>	Computes attachment score (Sec V)
$n$	Num of candidate parts	<i>ShapeFit()</i>	Computes shape score (Sec IV-B)
$m$	Num of parts in constructed tool	<i>Align()</i>	Aligns parts for computing P (Sec V)
$T$	${}^n P_m$ of parts in C	<i>ComputeIntersection()</i>	Computes P (Sec V)
$P$	Intersection point of parts	<i>ClosestAttachment()</i>	Returns closest attachments to P
$A$	Attachment points	<i>GraspSample()</i>	Samples tool grasps (Sec V-B)

TABLE I: Notation for terms and functions used in the paper

#### IV. SHAPE SCORING FOR TOOL CONSTRUCTION

There are two classes of approaches for shape scoring of parts: parametric (e.g., SQ) and non-parametric (e.g., ESF, SHOTC). Our core contribution in this paper concerns non-parametric shape scoring using ESF and SHOTC. In this section, we first discuss our prior work that used parametric shape scoring using SQs [19], and then further detail our current approach for non-parametric shape scoring.

##### A. Parametric shape scoring using SQ

Our prior work [19] takes as input a “reference tool” point cloud that the algorithm seeks to construct using the candidate parts. Thus, the output tool tries to match the shape of a specific input reference tool as closely as possible. SQs are used to represent the reference tool and candidate parts. SQs are represented by 13 parameters: 3 for scale in each dimension, 2 for shape variance, 3 for Euler angles, 2 for tapering parameters and 3 for the central point/mean. To find the best-fit SQ parameters, the authors perform non-linear optimization individually for each candidate object and reference tool part. The shape score is computed as the L1 norm between the SQ parameters of the candidate objects and the SQ parameters of the reference tool parts. The non-linear optimization required by this approach does not computationally scale as number of parts increases. Further, weights indicating relative importance of the different SQ parameters, need to be manually tuned to achieve desirable performance.

##### B. Non-parametric shape scoring using ESF and SHOTC

This section highlights one of the core contributions of this paper. Our current approach, takes as input a “reference action” as opposed to a reference tool point cloud, allowing a broader range of tools to be constructed without conforming to the shape specifications of a particular reference tool. Our algorithm seeks to accomplish a target action as opposed to match a reference tool. Additionally, in the evaluation, we compare shape scoring performances of ESF and SHOTC to SQs, and highlight the scalability and parameter tuning challenges associated with the parametric approach.

Given a reference action as input, the shape scoring module seeks to predict the shape fitness of the parts and output a score. We consider tools to have action parts and grasp

parts<sup>2</sup> and begin by training independent neural networks corresponding to specific reference actions. In this work, we utilize five different reference actions: “Hit”, “Contain/Scoop”, “Screw”, “Flip” and “Squeegee”. Additionally, we consider a supporting function: “Handle”, which refers to the tools’ grasp part. Thus, the first five correspond to the action parts of the tool which is combined with “handle” or grasp part for constructing the final tool. Each neural network takes as input the shape descriptor for a part (ESF or SHOTC) and outputs a binary label indicating whether the part is suitable for the function. The NN architectures used for ESF and SHOTC both consist of a single hidden layer, with 426 and 235 units respectively. We use ReLU activation, sigmoid in the final layer, and Stochastic Gradient Descent for training. Note that, ESF is a 640-D vector and SHOTC is a 352-D vector.

For training each action network, we generated a dataset consisting of 3D point clouds from the ToolWeb dataset [1] and other online sources<sup>3</sup>. The models were segmented<sup>4</sup> into their action and grasp parts – e.g., a hammer from the dataset is segmented to its hammer head (action part) and handle (grasp part) – and each part was added to the dataset of its respective type. This process resulted in six datasets corresponding to the five action types and the ‘handle’ set, with roughly 32 models per class, for a total of 196 point clouds. For each point cloud, we extract ESF and SHOTC features and train each neural network. The point clouds belonging to the particular class/function are treated as positive examples and point clouds from other classes are used as negative examples during training (corresponding 10-fold cross validation accuracies shown in Table II). For new actions, additional networks can be trained independently without affecting other networks.

To calculate a shape fit score, the scoring function *ShapeFit()* is provided with the reference action and a tuple of candidate parts  $T_i$ . Ordering of parts within the tuple indicates correspondence to action or grasp parts. Let  $\mathcal{K}$  denote the set of parts in  $T_i$  that are candidates for the action parts of the final tool, and let  $T_i - \mathcal{K}$  be the set of candidate grasp parts. Then the shape score  $e_{shape}^{T_i}$  is computed by *ShapeFit()* (Line 4) using the trained networks as:

$$e_{shape}^{T_i}(action) = \prod_{j \in \mathcal{K}} p(action|T_{ij}) \prod_{j \in T_i - \mathcal{K}} p(handle|T_{ij})$$

where,  $p$  is the prediction confidence of the corresponding network. Thus, we combine prediction confidences for all action parts and grasp parts. For example, if the specified action is ‘hit’ and  $T_i$  consists of two objects ( $t_1, t_2$ ), then  $e_{shape}^{T_i} = p(hit|t_1) * p(handle|t_2)$ . In the following section, we describe the computation of attachment score.

## V. ATTACHMENTS SCORING FOR TOOL CONSTRUCTION

The shape score discussed above indicates visual appropriateness of parts, but does not indicate whether the parts *can* be attached. To evaluate whether the parts can be attached

<sup>2</sup>This covers the vast majority of tools [18, 2]

<sup>3</sup>We used 3dWarehouse and tf3dm

<sup>4</sup>Approach used [https://github.com/pauloabelha/batch\\_segmentation](https://github.com/pauloabelha/batch_segmentation)

ACTION	ESF	SHOTC
Hit	95%	80.5%
Screw	95%	53.3%
Flip	97.5%	69.1%
Squeegee	70.%	95%
Contain	92.4%	93%
Handle	90.6%	83.3%

TABLE II: Cross-validation accuracies for ESF and SHOTC

---

### Algorithm 1: Attachment Fit (Sec V)

---

```

input : candidate tool parts  $T_i$ , attachment type  $t_{att}$ 
output:  $e_{att}^{T_i}, A_{closest}$ 

1  $e_{att}^{T_i} = 0, A_{closest} = []$ 
2  $T'_i = Align(T_i)$ 
3  $P = ComputeIntersections(T'_i)$ 
   // Compute A based on attachment type
4 if  $t_{att} = 'pierce'$  then
5   | if isPierceable( $T_i$ ) then
6     |   |  $A = P$ 
7   | else
8     |   |  $A = \emptyset$ 
9   | end
10 else if  $t_{att} = 'grasp'$  then
11   |  $A = GraspSample(T_i)$ 
12 else if  $t_{att} = 'magnetic'$  then
13   |  $A = userInput(T_i)$  // Predefined locs
14 else
15   |  $A = \emptyset$ 
16 end
17 if  $A \neq \emptyset$  then
18   | foreach  $t_i \in T'_i, s_j \in t_i$  do
19     |   |  $a = ClosestAttachments(P, s_j, A)$ 
20     |   |  $e_{att}^{T_i} \pm \|P, a\|$  // Euclidean dist to P
21     |   |  $A_{closest}.append(a)$ 
22   | end
23 else
24   |  $e_{att}^{T_i} = \infty$ 
25   | return  $e_{att}^{T_i}, P$ 
26 end
27  $\gamma = -max(e_{att}(T_i))$  // normalizer
28 return  $e_{att}^{T_i}/\gamma, A_{closest}$ 

```

---

appropriately to accomplish the specified reference action, we compute an attachment score. The attachment scoring algorithm is shown in Algorithm 1 (*AttachmentFit()*). Attachment scoring begins by aligning the components of the candidate tool  $T_i$  in a configuration consistent with prototypical tools used for the specified action (*Align()*, line 2). In order to retrieve this configuration, we sample one random tool from the dataset used in Sec IV-A, corresponding to the specified action. Further, we use Principal Component Analysis (PCA) to orient the part point clouds in  $T_i$  with

respect to the example tool, resulting in a set of alignments  $T'_i$ . We approximate the intersections of the point clouds in each alignment by calculating the centroid of closest points between the point clouds (*ComputeIntersection()*, line 3). The resultant set of centroids,  $P$ , is the candidate list of attachments we want to make. Depending on the attachment type  $t_{att}$ , the set of attachment locations  $A$  is computed (line 4-16). The attachment score  $e_{att}^{T_i}$  is computed based on the Euclidean distance between points in  $P$  and the closest attachment points in  $A$ , on each part  $s_j$ , in each alignment  $t_i \in T'_i$  (*ClosestAttachment()*, lines 18-22). The resulting score,  $e_{att}$ , is normalized (by  $\gamma$ ) (Line 27, Alg 1). The negative normalizer ranks lower  $e_{att}$  as better. If a part in  $t_i \in T_i$  is known to have no attachment points,  $e_{att}^{T_i} = \infty$ .

The set of attachment points  $A$  is required to compute  $e_{att}$ . Hence, we first identify the attachment points by considering three different types of attachments in order of reducing complexity: ‘pierce’, ‘grasp’ and ‘magnetic’ attachments. In the case of pierce and grasp attachments, we assume that the capabilities of the acting tool is known ( $t_{att}$  is known). That is, objects with pierce capability (screwdrivers and sharp pointed objects) and objects with grasp capability (pliers and tongs) are known a-priori, although these can be identified using existing affordance learning approaches [9]. In the following subsections, we describe how  $A$  is computed for each attachment type (lines 4-16, Alg 1).

#### A. Pierce Attachment

For determining pierceability, we use the SCiO which is a hand-held spectrometer (shown in Fig 4). The SCiO senses object material properties, returning a 331-D real-valued vector; in prior work the SCiO was shown to achieve 94.6% accuracy on material recognition tasks[11]. To predict pierceability, given a dataset of SCiO measurements from an assortment of objects, we train a model to output a binary label indicating material pierceability. We assume homogeneity of materials for pierceability prediction.

For our model, we use a neural network with a single hidden layer of 256 units and binary output layer. We used the Adam optimizer with ReLU activation layer, and a sigmoid in the final layer. To train our model, we used an existing dataset<sup>5</sup>, SMM50, with spectrometer readings for 4 classes of materials: plastic, wood, metal and foam. For each material class, 12 different objects were used with 50 samples collected per object from different locations of the object. This results in a total of 600 spectrometer readings per class. For each, we provide the pierceability labels. In our case, only 150/600 spectral readings correspond to pierceable objects. Our model yields an accuracy of 98% with leave-one-out cross validation on the SMM50 dataset.

To determine the attachment score during tool construction for the input  $T_i$ , the SCiO sensor is used to scan the objects and the corresponding spectral reading is passed to the classifier. If the output label is zero (line 5,  $isPierceable(T_i) = 0$ ,

$A = \emptyset$  since pierce attachment is not possible. If pierceable,  $A = P$ , assuming homogeneity of material properties allowing the objects to be configured at the desired location.  $A$  is then used to compute  $e_{att}$  based on Euclidean proximity to  $P$ .

#### B. Grasp Attachment

Grasp attachment is defined as using one object to grasp/hold another object to extend the robot’s reach (e.g., grasping a bowl with pliers). We model the grasping tool (pliers or tongs) as an extended robot gripper, allowing the use of existing robot grasp sampling approaches [27, 17, 31] for computing locations where the tool can grasp objects. In particular, we use the approach discussed in [27] that outputs a set of grasp locations, given the input parameters reflecting the attributes of the pliers/tongs used for grasping. We cluster the grasp locations (using Euclidean metric) to identify unique grasps. As described in [27], without any additional training, the geometry-based grasp sampling approach achieves an accuracy of 73%. To further improve the accuracy, it is possible to train an object-specific model to identify valid grasps. A key challenge with using a pre-trained model is the need to retrain it for every newly encountered pliers/tongs with differing parameters. This can be inefficient in terms of computational time and resources. Hence, we use the geometry-based grasp sampling without any object-specific refinement.

To determine the attachment score during tool construction for the input  $T_i$ , grasps are sampled for the objects (Line 11, *GraspSample( $T_i$ )*) using the existing grasp sampling algorithm<sup>6</sup>. Once sampled, the resultant grasps are returned as potential attachment locations  $A$ .  $A$  is then used to compute  $e_{att}$  based on Euclidean proximity to  $P$ .

#### C. Magnetic Attachment

We assume the locations of magnets to be provided or predefined, and simply compute the  $e_{att}$  score based on the Euclidean proximity of user input attachment locations (Line 13, *userInput( $T_i$ )*) to  $P$ . However, as described in [19], it is also possible to perform magnetic attachments via exploration if they are not predefined.

## VI. PARTS RANKING COMPUTATION

Given the shape score from Section IV, and the attachment score from Section V, we compute the aggregate final score as a sum of  $e_{shape}^{T_i}$  and the normalized  $e_{att}^{T_i}$ . The list of candidate part configurations are then sorted by their associated combined scores, from highest to lowest. The resultant ranking is then used by the robot to construct the final tool.

## VII. EXPERIMENTAL VALIDATION AND RESULTS

In this section, we describe our experimental setup and present our results alongside each evaluation. We validate our approach on the construction of tools for the five different actions, encoded as textual inputs: ‘hit’, ‘scoop/contain’, ‘flip’, ‘screw’ and ‘squeegee’. Each tool consists of two components ( $m = 2$ ) corresponding to the action part (‘hit’,

<sup>5</sup>Dataset available at <https://github.com/Healthcare-Robotics/smm50>

<sup>6</sup>Implementation at <https://github.com/atenpas/gpg> based on [27]

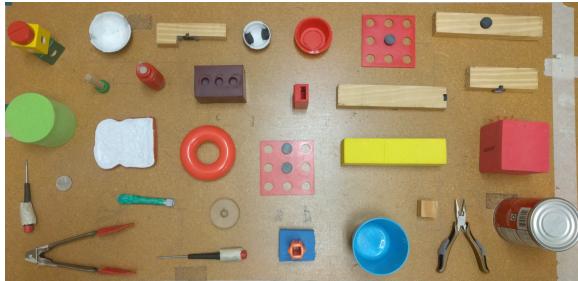


Fig. 3: The 30 objects used for experimental validation.

‘scoop/contain’, ‘flip’, ‘screw’, ‘squeegee’) and grasp part (‘handle’). The performance of the algorithm is evaluated for each tool in terms of the final ranking output by our algorithm. The tool models used to compute the desired attachment location  $P$ , is acquired from the ToolWeb dataset [2]. Our experiments seek to validate three key aspects of our approach:

- 1) *Attachment scoring evaluation*: Performance of the different attachment predictors, namely, pierce and grasp predictors;
- 2) *Comparison of parametric and non-parametric shape scoring*: Performance of ESF, SHOTC and SQ representation of tools for shape scoring;
- 3) *Final tool ranking evaluation*: Performance of the overall pipeline in terms of final tool ranking for the five actions described above. We evaluate this for three cases: Considering only shape scoring, only attachment scoring, combined shape and attachment scoring.

For all our following experiments, we use a test set consisting of 30 previously unseen candidate parts for tool construction (shown in Fig 3). These objects consist of metal (8/30), wood (8/30), plastic (9/30) and foam (5/30) objects. Only 4 of the 5 foam objects are pierceable, while the remaining 26 objects are non-pierceable. Fig 4 shows a sample experimental setup and steps involved in the robot tool construction. During tool construction, the robot begins by sensing the material properties of the objects and combines the attachment score and shape score for a final ranking of parts<sup>7</sup>. To overcome manipulation and perception challenges that are beyond the scope of this work, the available parts were spaced apart and oriented to facilitate grasping.

#### A. Attachment Scoring

In this section, we evaluate the performance of the pierceability and grasp sampling prediction. For this evaluation, each attachment predictor is tested on the full set of 30 objects and we report the accuracy indicating the number of objects for which the pierceability or graspability is correctly identified.

The results for the 30 test objects are shown in Fig 5a. Non-pierceable objects are classified correctly in 88% of the cases, and a 100% of all the pierceable objects are correctly categorized as pierceable, resulting in an overall accuracy of 90% on the test set for pierceability prediction.

<sup>7</sup>See attached supplementary video for demonstrations of our approach on the robot

For grasp attachment, the results are shown in Fig 5b, with an overall accuracy of 67% on the test of 30 objects. There are several false positives which potentially affects the ranking of the correct combinations. However, during the tool construction phase, the false positives are eliminated when the robot attempts the actual construction (see video for example), albeit resulting in a greater number of tool construction attempts on the robot. In future, grasp prediction could be improved if needed by using a model pre-trained on the candidate objects [27].

#### B. Parametric and non-parametric shape scoring

In this section, we compare the performance of the parametric and non-parametric shape scoring approaches. We use five different sets of four objects (chosen from the 30) for each of the five tools, and report the average results (total  $5 \times 5$  cases with four candidate objects per case). In each set, we include at least one “correct” combination of objects and the remainder of the objects are chosen randomly. The correct combination is determined based on human assessment of the parts. We rank the parts using *only* the non-parametric (ESF/SHOTC using reference action) or parametric (SQ using reference tool) shape scoring, without attachment scoring.

The metrics used in this evaluation are i) the final ranking of the correct combinations, and ii) the computation time. We would like the correct combination to be ranked as high as possible, ideally ranked at 1, indicating that it would be the first part combination the robot will attempt to construct. We report the average rank of the correct combination for each tool (average of 5 builds), the number of builds for which the correct combination was ranked within the top 5 ranks (hits@5), the average number of possible configurations of parts, and the average total computation time. The number of part configurations highlight the complexity of the state space and is also used to compute the percentage of total part combinations explored.

The results are shown in Table III. As can be seen by the average rank and hits@5 metrics, SQ and ESF perform somewhat better than SHOTC, although no method dominates all others. However, ESF has significantly better run time performance compared to SQ (average 0.178 seconds for ESF compared to 12.96 seconds for SQ). Hence, the learning framework is more scalable as the number of parts increase, mitigating the impact of a combinatorial search space. Overall, since ESF outranks SQ and SHOTC on three out of five tools, and is computationally significantly faster, we use only ESF in our final evaluation in the following section.

#### C. Final Tool Ranking

In this section, we evaluate our overall tool construction pipeline, combining both shape and attachment reasoning. We use the same sets of objects and evaluation metrics as used in the previous section, again with five builds per reference action. We compare the performance of four part ranking approaches: using only shape scoring with ESF (*Shape*), using only attachment scoring (*Att*), using combined attachment and

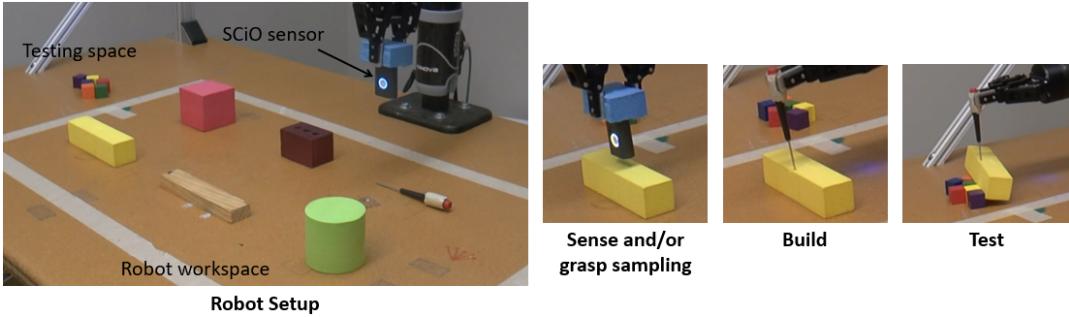


Fig. 4: Image showing robot setup and steps involved in a typical tool construction cycle. In the case of pierce attachment, the robot uses the SCiO sensor to sense material properties and in case of grasp attachment, the robot samples valid grasps for the object. The robot then builds the tool and tests it by performing the reference action with the tool.

REFERENCE TOOL															
REFERENCE ACTION	HIT			SCOOP/CONTAIN			FLIP			SCREW			SQUEEGEE		
REPRESENTATION	SQ	SHOTC	ESF	SQ	SHOTC	ESF	SQ	SHOTC	ESF	SQ	SHOTC	ESF	SQ	SHOTC	ESF
AVERAGE RANK	3	5	<b>3</b>	<b>3</b>	5	4	<b>2</b>	3	<b>2</b>	6	10	<b>6</b>	7	<b>4</b>	5
HITS@5	4/5	3/5	<b>5/5</b>	<b>5/5</b>	3/5	4/5	<b>5/5</b>	5/5	<b>5/5</b>	2/5	0/5	<b>3/5</b>	2/5	<b>4/5</b>	3/5
AVG. COMPUTATION TIME (seconds)	11.3	1.18	0.18	12.45	1.18	0.2	13.81	1.2	0.18	13.78	1.14	0.17	13.48	1.12	0.16

TABLE III: Performance of SQ, ESF and SHOTC (only shape scoring is used) averaged across the five builds for each of the five tools. The shown reference tools are used for SQ computations and reference actions for ESF and SHOTC; purple denotes the action part and green denotes handle. Shown in bold are the best performing representations for each action.

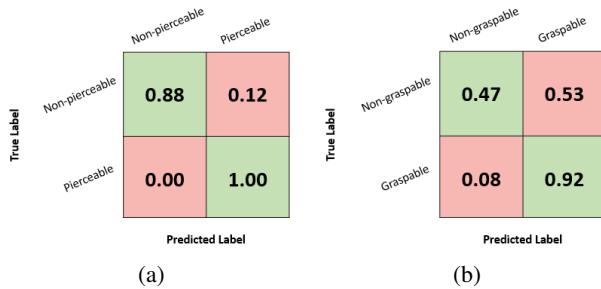


Fig. 5: Confusion matrixes for (a) pierceability prediction (90% accuracy), and (b) graspability prediction (67% accuracy), over the set of 30 objects.

shape scoring (*Shape + Att*), and using random ranking as a baseline (*Rnd*) to highlight the problem complexity.

The results of the evaluation are shown in Table IV and the corresponding tool constructions are shown in Table V. The random baseline achieves an average rank of 12, resulting in trials of 56% of candidate part pairs, on average, before finding a working tool solution. Only 24% of builds are completed in fewer than five tries (hits@5 metric).

In the ESF-only (*Shape*) condition, the algorithm achieves an average ranking of 4, and in 80% of trials the correct combination is ranked in the top five (hits@5). On average,

only 19% of all the part combinations are explored. These results demonstrate that shape information, even if used alone, is a useful metric for pruning the search space of potential part combinations.

In the attachment-only (*Att*) condition, the algorithm also receives an average ranking of 4, and in 92% of trials the correct tool combination is ranked in the top five (hits@5). On average, 18% of the total part combinations are explored. These results show that predicting which parts could be connected together, even without reasoning about the suitability of their shape, again leads to improved performance in the search for the correct part combination, especially in the number of cases the correct parts are ranked in the top five. In the combined shape and attachment (*Shape + Att*) condition, the algorithm achieves an average ranking of 2. In 100% of cases, the correct part combination is ranked in the top five (hits@5). On average, only 10.4% of the total space of part combinations is explored. These results highlight the complimentary nature of the shape and attachment metrics, and their effectiveness in consistently predicting the correct part combination.

#### D. Key findings

In this section, we highlight several important findings. First, the final approach was **efficient** in that, the robot explored only a small percentage of possible part combinations

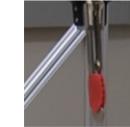
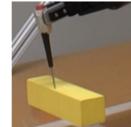
REF. ACTION	HIT				SCOOP/CONTAIN				FLIP				SCREW				SQUEEGEE			
SCORING STRATEGY	Shape (ESF)	Att	Shape + Att	Rnd	Shape (ESF)	Att	Shape + Att	Rnd	Shape (ESF)	Att	Shape + Att	Rnd	Shape (ESF)	Att	Shape + Att	Rnd	Shape (ESF)	Att	Shape + Att	Rnd
AVERAGE RANK	3	3	<b>2</b>	11	4	3	<b>2</b>	9	2	4	<b>1</b>	19	6	6	<b>4</b>	10	5	3	<b>2</b>	12
HITS@5	5/5	5/5	<b>5/5</b>	1/5	4/5	4/5	<b>5/5</b>	2/5	5/5	5/5	<b>5/5</b>	0/5	3/5	4/5	<b>5/5</b>	3/5	3/5	5/5	<b>5/5</b>	0/5
PROPORTION OF SPACE EXPLORED	0.17	0.17	<b>0.12</b>	0.65	0.2	0.15	<b>0.1</b>	0.45	0.07	0.13	<b>0.03</b>	0.63	0.24	0.24	<b>0.16</b>	0.4	0.27	0.2	<b>0.11</b>	0.67
AVG. NUMBER OF POSSIBLE CONFIGS	17				20				30				25				18			
EXAMPLE TOOL CONSTRUCTED WITH PARTS AND ATTACHMENT TYPE USED																				
	Magnetic attachment: two wooden pieces				Grasp attachment: Red bowl and tongs				Grasp attachment: Flat piece and tongs				Grasp attachment: Coin and tongs				Pierce attachment: Foam block and screwdriver			

TABLE IV: Performance of the *Shape*, *Att*, *Shape+Att*, and *Rnd* conditions for each of the five target actions, averaged over five builds. Shown in bold is the best performing strategy. One example built tool is shown for each action.

(average 10.4%), on average requiring only *two* construction attempts per tool. The computational time of the shape modeling component was only 0.178 seconds, indicating that this technique will scale well to larger domains. Second, we found that **spectrometer readings were very effective** in predicting the pierceability of objects, yielding an overall accuracy of 90%. In future work, we hope to explore other uses of spectrometer data in the tool building context. Third, we found that **shape reasoning with ESF** outperformed SQ and SHOTC in terms of average ranking and computation time. Fourth, either shape or attachment reasoning alone already provided significant guidance in tool creation. However, **combined shape and attachment reasoning** led to the most efficient performance. To the best of our knowledge, this is the first work to demonstrate physical tool construction utilizing multiple attachment types.

### VIII. CONCLUSION AND FUTURE WORK

In this work, we have contributed a novel reasoning framework for tool construction based on shape and attachment reasoning, which we have demonstrated on the construction of tools for five reference actions. Additionally, we compared the performance of ESF, SHOTC and SQ shape scoring, and presented an approach to determine material pierceability from spectroscopy. In contrast to our prior work on tool construction [19], our improved approach uses a target action and richer attachment modalities to reason about and construct a broad range of tools. In all cases, the approach efficiently discovered working combinations, exploring only a small percentage of all possible part combinations.

In future work, a number of changes can be made to further improve the performance of the system. For example, we observed cases in which shape scoring produced incorrect ranking because the RGBD sensor captured only a partial point cloud of an object. In the video, this can be seen during the construction of a scoop, when the robot incorrectly chooses a green foam cylinder (which is not a concave object), as

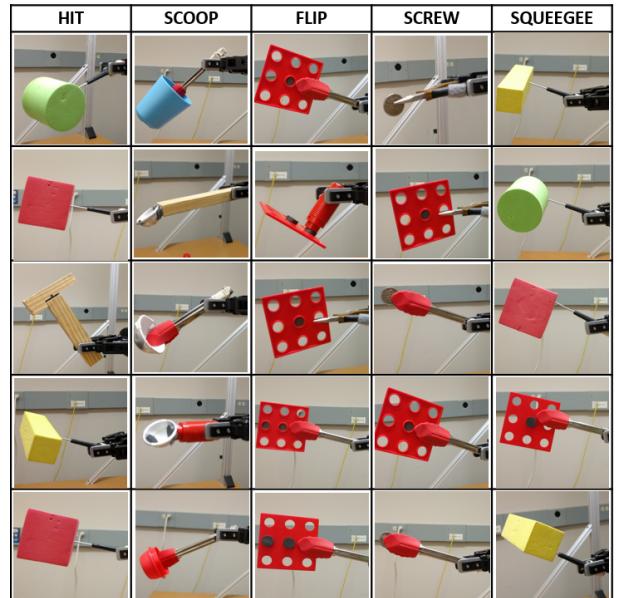


TABLE V: Table showing each of the tools constructed for the five reference actions. Note that a small number of experiments led to the creation of similar tools due to the availability of parts that could be connected.

opposed to a red bowl since the RGBD sensor only receives a partial view of the green foam. Future work can address such problems through active perception. Additionally, our future work will address a key limitation of our current approach that the number of parts utilized for constructing a tool equals number of tool parts i.e, there is a one-to-one correspondence between candidate parts and tool parts. Finally, additional material properties, such as mass and density, can be incorporated into the reasoning framework to further improve performance.

### ACKNOWLEDGMENTS

This work is supported in part by NSF IIS 1564080 and ONR N000141612835.

## REFERENCES

- [1] Paulo Abelha, Frank Guerin, and Markus Schoeler. A model-based approach to finding substitute tools in 3d vision data. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 2471–2478. IEEE, 2016.
- [2] Paulo Abelha Ferreira and Frank Guerin. Learning how a tool affords by simulating 3d models from the web. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems (IROS 2017)*. IEEE Press, 2017.
- [3] Aitor Aldoma, Zoltan-Csaba Marton, Federico Tombari, Walter Wohlkinger, Christian Potthast, Bernhard Zeisl, Radu Bogdan Rusu, Suat Gedikli, and Markus Vincze. Tutorial: Point cloud library: Three-dimensional object recognition and 6 dof pose estimation. *IEEE Robotics & Automation Magazine*, 19(3):80–91, 2012.
- [4] Alan H Barr. Superquadrics and angle-preserving transformations. *IEEE Computer graphics and Applications*, 1(1):11–23, 1981.
- [5] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Material recognition in the wild with the materials in context database. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3479–3487, 2015.
- [6] Christopher D Bird and Nathan J Emery. Insightful problem solving and creative tool modification by captive nontool-using rooks. *Proceedings of the National Academy of Sciences*, 106(25):10370–10375, 2009.
- [7] Christophe Boesch and Hedwige Boesch. Tool use and tool making in wild chimpanzees. *Folia primatologica*, 54(1-2):86–99, 1990.
- [8] Dongkyu Choi, Pat Langley, and Son Thanh To. Creating and using tools in a hybrid cognitive architecture. 2018.
- [9] Thanh-Toan Do, Anh Nguyen, and Ian Reid. Affordancenet: An end-to-end deep learning approach for object affordance detection. In *International Conference on Robotics and Automation (ICRA)*, 2018.
- [10] Can Erdogan and Mike Stilman. Planning in constraint space: Automated design of functional structures. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1807–1812. IEEE, 2013.
- [11] Zackory Erickson, Nathan Luskey, Sonia Chernova, and Charlie Kemp. Classification of household materials via spectroscopy. *IEEE Robotics and Automation Letters*, 2019.
- [12] Tesca Fitzgerald, Ashok K Goel, and Andrea L Thomaz. Representing skill demonstrations for adaptation and transfer. In *AAAI Symposium on Knowledge, Skill, and Behavior Transfer in Autonomous Robots*, 2014.
- [13] Paweł Gajewski, Paulo Ferreira, Georg Bartels, Chaozheng Wang, Frank Guerin, Bipin Indurkhya, Michael Beetz, and Bartłomiej Sniezynski. Adapting everyday manipulation skills to varied scenarios. *arXiv preprint arXiv:1803.02743*, 2018.
- [14] Diane Hu, Liefeng Bo, and Xiaofeng Ren. Toward robust material recognition for everyday objects. In *BMVC*, volume 2, page 6. Citeseer, 2011.
- [15] Thony B Jones and Alan C Kamil. Tool-making and tool-using in the northern blue jay. *Science*, 180(4090):1076–1078, 1973.
- [16] Martin Levihn and Mike Stilman. Using environment objects as tools: Unconventional door opening. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 2502–2508. IEEE, 2014.
- [17] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436, 2018.
- [18] Austin Myers, Ching Lik Teo, Cornelia Fermüller, and Yiannis Aloimonos. Affordance detection of tool parts from geometric features. In *ICRA*, pages 1374–1381, 2015.
- [19] Lakshmi Nair, Jonathan Balloch, and Sonia Chernova. Tool macgyvering: Tool construction using geometric reasoning. In *International Conference on Robotics and Automation (ICRA)*, 2019.
- [20] Muhammad Asif Rana, Mustafa Mukadam, S Reza Ahmadzadeh, Sonia Chernova, and Byron Boots. Towards robust skill generalization: Unifying learning from demonstration and motion planning. In *Conference on Robot Learning*, pages 109–118, 2017.
- [21] Vasanth Sarathy and Matthias Scheutz. The macgyver test-a framework for evaluating machine resourcefulness and creative problem solving. *arXiv preprint arXiv:1704.08350*, 2017.
- [22] Markus Schoeler and Florentin Wörgötter. Bootstrapping the semantics of tools: Affordance analysis of real world objects on a per-part basis. *IEEE Transactions on Cognitive and Developmental Systems*, 8(2):84–98, 2016.
- [23] Gabriel Schwartz and Ko Nishino. Recognizing material properties from images. *arXiv preprint arXiv:1801.03127*, 2018.
- [24] Mike Stilman, Munzir Zafar, Can Erdogan, Peng Hou, Saul Reynolds-Haertle, and Gregory Tracy. Robots using environment objects as tools the macgyver paradigm for mobile manipulation. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2568–2568. IEEE, 2014.
- [25] Dietrich Stout. Stone toolmaking and the evolution of human culture and cognition. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 366(1567):1050–1059, 2011.
- [26] Dietrich Stout and Thierry Chaminade. The evolutionary neuroscience of tool making. *Neuropsychologia*, 45(5):1091–1100, 2007.
- [27] Andreas ten Pas, Marcus Gualtieri, Kate Saenko, and Robert Platt. Grasp pose detection in point clouds. *The International Journal of Robotics Research*, 36(13-14):

- 1455–1473, 2017.
- [28] Nicholas Toth, Kathy D Schick, E Sue Savage-Rumbaugh, Rose A Sevcik, and Duane M Rumbaugh. Pan the tool-maker: investigations into the stone tool-making and tool-using capabilities of a bonobo (*pan paniscus*). *Journal of Archaeological Science*, 20(1):81–91, 1993.
  - [29] Handy Wicaksono and Claude Sammut1 Raymond Sheh. Towards explainable tool creation by a robot. In *IJCAI-17 Workshop on Explainable AI (XAI)*, page 63.
  - [30] Walter Wohlkinger and Markus Vincze. Ensemble of shape functions for 3d object classification. In *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*, pages 2987–2992. IEEE, 2011.
  - [31] Philipp Zech and Justus Piater. Grasp learning by sampling from demonstration. *arXiv preprint arXiv:1611.06366*, 2016.