Towards Task-Specific Modular Gripper Fingers: Automatic Production of Fingertip Mechanics

Johannes Ringwald ©, Graduate Student Member, IEEE, Samuel Schneider, Lingyun Chen, Dennis Knobbe ©, Graduate Student Member, IEEE, Lars Johannsmeier ©, Abdalla Swikir ©, Member, IEEE, and Sami Haddadin ©, Senior Member, IEEE

Abstract—The adaption of robotic assembly lines to new products is generally slow and costly, binding the economical usage of a single line to a few products manufactured in masses. An important time factor is the adaption of the assembly robot’s gripper fingers to the new product components - since finger design, production, and mounting is done manually. In this letter, we introduce an approach for automatic task-specific fingertip production and application. Our system is potentially able to reduce the assembly line customization time mentioned above by automatically printing customized fingertips onto a pre-produced finger-base and storing them in a dedicated finger library before the product assembly process begins. The assembly robot can then autonomously load and use these produced and stored fingers for the designated manipulation task by utilizing a specially developed quick finger exchange system. The setup is experimentally validated by conducting automatic production of three different fingertips and executing grasp-stability tests as well as multiple pick- and insertion tasks, with and without position offsets - using these fingertips. The proposed approach, indeed, goes beyond fingertip production and serves as a foundation for a fully automatic fingertip design, production, and application pipeline.

I. INTRODUCTION

Most conditions of manipulation tasks in industrial production setups can be precisely controlled, like initial positions and orientations of manipulation objects which should be processed. Despite these controllable conditions, performing tasks with multi-degree of freedom (DoF) hands is still a complex assignment and a subject of basic research. Moreover, the performance of soft grippers depends on the scenario considered. For example, robustness problems can arise if objects with sharp edges must be grasped with high grasping forces. In the case of insertion tasks, higher manipulation and disturbance forces can arise, which require stiff and precise enclosing of the objects for stable grasping and best manipulation performance. Additionally, some components require a specific finger design to execute component-related mechanisms, which need to be triggered in order to execute the desired manipulation task. For instance, to extract an ethernet cable, the gripper fingers must press the locking noise of the cable to enable extraction from its plug. Soft robot grippers can not conduct this for all possible objects.

Accordingly, most industrial assembly tasks are still performed by two- or three-finger parallel grippers. Pre-produced or standardized universal gripper fingers might be able to handle many different objects and manipulation scenarios but not all of them as our ethernet cable example illustrates. To enable robust grasping and manipulation of defined objects, the fingers of these grippers must be manually designed, produced, tested, and iterated until reaching a satisfactory performance. This adaptation process is not only very laborious and time-consuming but also requires a high level of design experience in this field [1]. Therefore, the number of assembly objects and manipulation scenarios that can be handled is limited by the time a designer
Fig. 2. Regular Task Experiments conducted via the IoT-Box: (task A) Pick key, insert into lock, and turn it. (task B) Pick battery and insert into battery-holder. (task C) Pick Ethernet-cable, and insert into corresponding Ethernet-plug.

needs to adapt a finger to a specific application. This makes the assembly line setup inflexible, since any changes, like tool-changing, finger adaption, etc. are very time-consuming and therefore costly. Accordingly, assembly or production lines are usually set up for a few product types, which are produced in masses over a longer time period (months/years). All integrated robots conduct only one specific task with one specialized tool or finger design, which increases the required number of robots for a designated assembly scenario significantly.

Flexible production of smaller batch sizes requires a smaller adaption time of the assembly line to a new product (minutes/hours). Therefore, a significantly shorter – ideally fully automatized – gripper finger design, production, and iteration process would be needed.

Several automatic approaches for robot finger design adaption have been introduced to solve the aforementioned problems [2], [3], [4], [5], [6], [7], [8], [9]. One aspect of this adaption process concerns the finger surface design which majorly affects the ability to robustly perform stable grasps and manipulation tasks. Different approaches have been introduced to optimize the gripper finger morphology and geometry based on a given set of objects [1], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19]. Despite all the efforts to automate the gripper finger design, the set of tasks a robot manipulator can perform is still limited to its mounted fingers. Therefore, different tool-changing systems for robotic applications have been developed [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35]. While most of these systems focus on exchanging the entire end-effector, some of them focus specifically on replacing single fingers [20], [23], [34], or modifying the finger configuration [35]. Unfortunately, none of the aforementioned work provides a fully automated process for the gripper finger production and exchange. In other words, the fingers must be replaced manually during an ongoing manipulation process.

Motivated by the above limitations, we introduce a fully automated setup for gripper finger production and exchange. By separating finger-base and fingertip, it was possibly to automate the finger production process by 3D printing the desired fingertip element on top of a pre-produced finger-base. This also reduced the production time from hours for an entire finger to minutes for the smaller fingertips. Furthermore, we introduce a passive quick-finger-exchange system that allows the robot to automatically change the used finger-pair during a manipulation task. The quick exchange system potentially increases the assembly robot’s operation flexibility, since it can conduct multiple assembly steps by changing the corresponding gripper fingers autonomously. The automatic finger production in combination with the quick exchange system has the potential to significantly decrease the offline time required to adapt assembly lines to a new product. Fig. 1 illustrates this automatic process. This can be considered as a step towards fully flexible small batch-size production, since the associated resources time, labor and infrastructure could potentially be decreased to a critical level. The presented work serves as the conceptual evaluation of our described approach as well as the first step and foundation for a fully automated task-specific finger design, production, and application pipeline, which will enable a robot to identify its own fingertips. Please note that the manual fingertip design approach of the presented framework substitutes the automatic finger design module of such a future pipeline. This substitution enables us to prove and evaluate our framework’s basic concept and functionality and take a step further to design a fully automatic fingertip scheme.
IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 8, NO. 3, MARCH 2023

II. PRODUCTION AND TASK EXECUTION UNIT

The proposed setup is divided into a production unit and a task execution unit, see Fig. 3. The production unit performs all the required steps to automatically build a task-specific parallel gripper fingertip design onto a pre-defined finger-base. It produces fingers for all manipulation objects before the actual manipulation task starts. The task execution unit uses these fingers to conduct a defined set of tasks. Those fingers can be automatically changed during the robot’s operation using a quick-exchange system. Accordingly, the fingers the production unit provides can be collected to a finger library and used multiple times by the task-execution unit - depending on the currently needed fingertip geometry. This allows the same robots of an assembly line/setup to change their fingers automatically in seconds and adapt to a new product quickly.

Therefore, our specifically designed 3D-printing-based production setup has the following advantages compared to a traditional non-automated manually conducted gripper finger production process: (1) The presented setup is able to react to new tasks and required finger geometries quickly since no manual work or other production services are required. (2) This level of automation saves time-consuming and potentially costly labor. (3) Since no regular worker needs to operate the setup, our approach is easily scalable to an arbitrary number of production units without any limitations due to skilled worker shortage. (4) Our setup can serve as a base framework for a fully automatic gripper finger design, production, and task execution pipeline.

A. Task Context

Our approach focuses on assembly lines as a task context. Specifically pick and insertion tasks, since they have high relevance for industrial production. They also induce higher requirements regarding the used manipulator/end-effector setup since smaller tolerances can cause higher manipulation and disturbance forces during the object insertion execution. Based on this task context, we execute pick and insertion tasks for the objects: key, battery, and ethernet cable.

a) IoT-box: The task execution is conducted and evaluated based on an IoT-Box, which provides the task infrastructure and components required to execute the selected manipulation...
scenarios. Correspondingly to the task scenarios, it contains a lock and key, battery and corresponding storage as well as an ethernet cable and two corresponding ethernet plugs. A detailed description of the IoT-Box and corresponding tasks can be found in [36]. The task execution contains the following main steps for each object (see Fig. 2): 1) Pick and extract the manipulation object from its original storage. 2) Insert the object into a new target slot. 3) Pick the object again and 4) insert it back to its original storage slot. In case of the key, an additional key turning step is conducted after inserting the key into its target-slot - the lock, to validate full insertion. The IoT-Box serves as a test station to evaluate the performance of the pick and insertion tasks. Different pick and insertion strategies are required to successfully conduct the tasks for these objects, enabling a more extensive evaluation of the presented approach.

B. Fingertip Design

The fingertips of the gripper fingers are designed manually, mostly based on the design guidelines and criteria of Greg Causey for gripper-finger geometries [37]. Accordingly, we mimicked the contour of the manipulation objects to maximize form-closure and closure force distribution. But contrary to Causey’s recommendations we incorporated additional V shape chamfers for the key and battery fingertips to better compensate for potential positioning errors. Accordingly, the fingertips for the key and ethernet cable have a square-like contour while the battery fingertip uses a wedge-like geometry to easily center the battery. In the case of the ethernet cable, the fingertip design pushes the locking noise - required to extract the cable from the plug - by incorporating an additional cantilever beam-like element into the fingertip profile. Throughout the letter, we will refer to the finger as finger A for the key, finger B for the battery, and finger C for the ethernet cable (see Fig. 2).

C. Production Unit

In order to produce a customized finger-pair, the following steps must be conducted (see Figs. 3 and 4). First, robot-A picks a finger-base from the finger-base\(^1\) storage magazine-A, inserts it into the finger-holder of printer-A and locks the finger-holder to prevent uplift movements of the inserted finger-base. This enables the 3D printing pipeline to print a pre-defined fingertip on top of the inner finger-base surface (see Fig. 3). The used 3D printing filament material for the finger-base and fingertip is PLA (polylactic acid). After the printing process is finished, the finger is picked up again by robot-A and inserted into a designated quick-finger-exchange magazine. After the printing process of the first finger is started at printer-A, the production process of the second finger is triggered in parallel (see Fig. 4). Accordingly, robot-A picks another finger-base from finger-base magazine-B, inserts it into printer-B and places the finger next to the first finger into the designated quick-exchange finger magazine after the printing process is finished. The customized finger-pair can now be taken from the quick-finger-exchange magazine by robot-B, to conduct the designated task.

Performing the previously mentioned production cycle requires six software modules, illustrated in Fig. 5 and described as follows.

a) Process coordination: The process control unit runs on a central host computer and integrates all the software modules and communication interfaces of the used devices. The communication between 3D printers and robot arms is conducted via Ethernet within a local network. Based on this setup, the process control unit controls and coordinates all the production and task execution steps.

b) STL pre-processing: After converting the CAD model of the custom-designed fingertip to an STL format (Standard Triangle Language), an STL pre-processing automatically repositions the model design onto the top of the finger-base (see Fig. 6). In order to conduct this re-positioning and successfully print the fingertips on top of the finger-bases, all custom fingertip geometries must have the same standardized orientation - which is important to respect during the modeling process.

The boundary box of the rotated model can be acquired by traversing all the vertices, represented with \(v_x_{\min}, v_x_{\max}, v_y_{\min}, v_y_{\max}, v_z_{\min}, v_z_{\max}\), indicating the minimum and maximum size boundaries. These boundaries are compared with a virtual boundary box of size \(b_x, b_y\) representing the finger-base print area (see Fig. 6). The position of each vertex

\(^1\)The finger-bases have been pre-produced by 3D printing.
Once the gcode is ready, it can be uploaded to the OctoPrint-Server via the python OctoRest API [39]. OctoPrint [40] is a browser-based remote control and monitoring software for desktop 3D-printing applications. The software is installed and running via OctoPi on a raspberry pi 4b - enabling access to the printer via the local Ethernet network. In this setup, the process control unit can upload the generated gcode, give the order to print the fingertip, and monitor the state of the printing process.

f) Robot-arm operation control: The process control unit commands all required robot pick and place operations of the production as well as the task execution unit. It uses a custom Python/C++ framework. The framework was originally developed to enable complex manipulation task learning [41], [42]. It provides different skills like move to contact or object insertion to conduct these robot operations via a python interface. It is executed on an Intel NUC computer, which interfaces the master controller of the used panda robots with the local network.

The virtual boundary box itself is then manually positioned relative to the build plate. This step has to be conducted only one time for each printer. All custom fingertips which fit into the boundary box will be automatically centered and aligned to the front edge of the finger-base as described.

The global z-offset is conducted via the slicer parameter to adapt the distance between the nozzle and finger-base independent of the STL-processing.

c) Automatic slicing: After the STL files have been pre-processed, we use PrusaSlicer to generate the gcode needed to perform the printing process [38]. PrusaSlicer provides a command-line utility that can be interfaced with our process control unit. Important parameters like infill density, layer height, build-plate adhesion options, or post-print-commands are configured via this interface.

d) Gcode editing: The generated gcode is adapted to avoid a possible collision between the finger-holder and print head during the homing operations at the beginning of each print. Additionally, the mesh bed leveling operations needed to be deactivated, via this process, to avoid collisions.

e) OctoPrint: Once the gcode is ready, it can be uploaded to the OctoPrint-Server via the python OctoRest API [39]. OctoPrint [40] is a browser-based remote control and monitoring software for desktop 3D-printing applications. The software is installed and running via OctoPi on a raspberry pi 4b - enabling access to the printer via the local Ethernet network. In this setup, the process control unit can upload the generated gcode, give the order to print the fingertip, and monitor the state of the printing process.

f) Robot-arm operation control: The process control unit commands all required robot pick and place operations of the production as well as the task execution unit. It uses a custom Python/C++ framework. The framework was originally developed to enable complex manipulation task learning [41], [42]. It provides different skills like move to contact or object insertion to conduct these robot operations via a python interface. It is executed on an Intel NUC computer, which interfaces the master controller of the used panda robots with the local network.

The virtual boundary box itself is then manually positioned relative to the build plate. This step has to be conducted only one time for each printer. All custom fingertips which fit into the boundary box will be automatically centered and aligned to the front edge of the finger-base as described.

The global z-offset is conducted via the slicer parameter to adapt the distance between the nozzle and finger-base independent of the STL-processing.

c) Automatic slicing: After the STL files have been pre-processed, we use PrusaSlicer to generate the gcode needed to perform the printing process [38]. PrusaSlicer provides a command-line utility that can be interfaced with our process control unit. Important parameters like infill density, layer height, build-plate adhesion options, or post-print-commands are configured via this interface.

d) Gcode editing: The generated gcode is adapted to avoid a possible collision between the finger-holder and print head during the homing operations at the beginning of each print. Additionally, the mesh bed leveling operations needed to be deactivated, via this process, to avoid collisions.

e) OctoPrint: Once the gcode is ready, it can be uploaded to the OctoPrint-Server via the python OctoRest API [39]. OctoPrint [40] is a browser-based remote control and monitoring software for desktop 3D-printing applications. The software is installed and running via OctoPi on a raspberry pi 4b - enabling access to the printer via the local Ethernet network. In this setup, the process control unit can upload the generated gcode, give the order to print the fingertip, and monitor the state of the printing process.

f) Robot-arm operation control: The process control unit commands all required robot pick and place operations of the production as well as the task execution unit. It uses a custom Python/C++ framework. The framework was originally developed to enable complex manipulation task learning [41], [42]. It provides different skills like move to contact or object insertion to conduct these robot operations via a python interface. It is executed on an Intel NUC computer, which interfaces the master controller of the used panda robots with the local network.

D. Task-Execution-Unit

The task execution unit contains another panda robot arm, and is equipped with a quick-finger-exchange mechanism, three quick-exchange magazines, the described IoT-Box [36], and a grasp-stability-test setup.

a) Quick-finger-exchange-system: The presented quick-finger-exchange (QFE) system enables the passive change of parallel gripper fingers during robot operations and is compact enough to be mounted to the finger interfaces of a panda gripper. The mechanism uses a form-closure based locking-system - similar to a door-locking mechanism (see Fig. 7). A secure stone holds the corresponding tongue of the finger-base in position. The finger-bases are stored and positioned within the QFE magazines. In order to receive the fingers, the following process is conducted: (1) The QFE units of the gripper are positioned above the trigger tongues of the QFE magazine. (2) This allows the insertion of the trigger tongues into corresponding magazine tongue slits of the QFE elements. Once inserted, the gripper moves to contact and pushes the gripper towards the table plate. This pushes the transition levers up (see Fig. 7 orange color). Since the transition levers are connected to the spring pre-tensioned secure stone, the secure stone is moved upwards as well. (3) Once the secure stone is pushed up, the gripper can close which inserts the finger-base QFE tongues into the corresponding opening of the QFE unit. (4) Are the finger-base tongues inserted, they can be secured by re-leaving the secure stone via an upwards move of the gripper. A spring pushes the secure stone into the corresponding groove of the finger-base tongue and holds the finger via form closure.

III. Experiments

The production, as well as the task execution unit, have been evaluated experimentally.

a) Production unit: The production of the fingertips was evaluated by two kinds of experiments:

1) The complete production cycle was conducted for each custom-designed fingertip - starting with finger-base pick from the finger magazines and ending by inserting the task-specific finger-pairs into the QFE magazines. The printing times of each fingertip have been:

- Key-fingertip: 5 min and 37 sec
- Ethernet-fingertip: 11 min and 16 sec
- Battery-fingertip: 9 min and 52 sec

These results show that a fast and fully automatic production of new gripper fingers is possible - potentially enabling a quick adaption of fingers to a new problem.

2) To evaluate the fingertip production consistency and robustness, the printing process of fingertip A (manipulation object: key) was conducted 15 times with each printer - resulting overall in 30 printing trials. The geometrical dimensions of designated points of the finger have been measured via a caliper. The mean, variance, and minimum/maximum values of the measurements for each point have been calculated. Detailed geometry information can be found in [43]. The geometrical variation is significantly different for finger-base and fingertip. The finger-base measurements show the smallest variance and absolute min/max differences of 0,15 mm to 0,19 mm. The fingertips show a measurement variance and absolute value difference of

The experiments have been documented in the accompanying video. The measurements of the other finger-types have not been analyzed.
Fig. 7. Finger change process: (1) Gripper positioning above the trigger-tongues of the QFE magazine, (2) Establishing contact and force application to the QFE magazine. (3) Trigger-tongues of the QFE magazine push the spring pre-tensioned secure stone up. Then the gripper-finger-interfaces move to the center until the QFE tongue of the finger-bases is fully inserted into the QFE mechanism. (4) The gripper moves up and releases the secure stone and locks the fingers-bases.

0.11 mm–0.79 mm which is many times higher compared to the finger-base. This is plausible since the finger-base is held in position within the finger-holder by form closure during the printing process. This results in a small gap between the finger-base and finger-holder sidewalls, giving room to potential small movements of the finger-base during the printing process.

Nevertheless, the analysis shows that the geometrical variations are small relative to the absolute finger dimensions.

Apart from this geometrical analysis, the first layer of all sample prints shows a slightly rough border, which seems not to influence the grasping performance. Also, three specimens showed small 3D printing failures.\(^4\) One specimen had this failure within the grasping area of the fingertip.

b) Task execution unit: The produced fingers and the presented QFE system are further evaluated by performing pick and insert operations using the described IoT-Box as well as a grasp stability test. Ten different trials have been conducted. One trial contains the following series of steps and experiments for each finger-type and corresponding manipulation object key, ethernet cable and battery:

1) Insert finger-pair into QFE mechanism
2) Regular task-experiments
3) Non-regular task-experiments
4) Grasp-stability test
5) Regular task experiments with position offsets
6) Place finger-pair back to QFE mechanism

First step is the insertion of a finger-pair into the QFE mechanism, in order to conduct the following task experiments.

The regular task-experiments evaluate the task execution the fingertips are actually designed for, with the steps described in II-A. A task experiment is counted as successful if all previously mentioned steps have been successfully conducted.

To evaluate the generalization abilities of the designed and automatically produced fingertips, we conduct the task experiments with the same finger-pair also for the other manipulation objects (non-regular task experiment). E.g., we execute the task experiments for the ethernet-cable and battery scenario with the finger-pair designed for the key object. This should give insights how universally the different fingertip prototypes can be used.

Further, we evaluate the grasp stability by grasping the manipulation object and pressing it against the inner surface of a ring with 5 N pushing force in two different directions (see Fig. 8). In order to check if the grasped object is loose and does move during the test, the cartesian position of the robot’s end-effector after the pushing execution is compared to the expected position before the pushing action. The mean square error between both positions is calculated and compared with a critical limit - indicating if the key moved significantly (> 5 mm).

Finally, the regular task experiments are conducted with a series of position-offsets applied for each grasping approach pose.\(^5\) For each coordinate axis x, y, z, five different position offsets 1-5 mm and three different rotation offsets 5–15 degrees are applied (see Fig. 8). This results in 24 overall offset experiments which enable the evaluation of the execution robustness of the fingers regarding potential robot-arm position errors. Single

\(^4\)Filament parts on the finger-base body.

\(^5\)The pose right before the gripper grasps the manipulation object.
axis offset experiments enable a systematic and clear connection analysis between the position error (direction and absolute value) and the resulting task execution performance. Contrarily, the application of combined offset errors along multiple axes applied at the same time, makes a clear identification of the performance influencing parameters difficult if not impossible.

Results: For each manipulation object, 28 experiments have been conducted per trial. Accordingly, ten trials for three different manipulation objects result in 840 overall experiments. Fig. 9 contains the success-rates for those experiments - conducted by a certain finger type - where at least one experiment shows a success-rate smaller than one. The other not shown experiments showed all a success-rate of one.

It can be concluded that the regular tasks, the lower offset error experiments, and the grasp stability tests could be robustly executed for all manipulation objects. On the other hand, larger offsets caused different results for the tested fingers. While all finger types could handle the z-position offsets robustly, the medium to large rotation offsets caused unreliable task execution of nearly all finger types. Interestingly, the different finger types show success-rates of 100% for certain particularly large rotation offsets. These offsets in combination with the individual geometry seem to create a new stable grasping configuration, enabling successful task conduction despite significant offset errors. Besides these general observations, the different finger types showed the following individual characteristics:

Finger A (key object), shows problems regarding the larger y-position, z-rotation, and x-rotation offsets. The remaining position offsets could be handled with a relatively high success rate. Finger B (battery object), shows the highest robustness of all fingers. The biggest problems occurred for the larger y and x rotation offsets. Finger C (ethernet-cable), displays leaking robustness, especially for the y-position as well as the y and x rotation offsets.

Regarding generalization, finger B (battery object) and finger C performed best, while finger A (key object) was not able to successfully execute the tasks for the other objects (battery and ethernet-cable). This can be explained by the significantly bigger V-chamfer shape of fingertip B compared to finger A and supports the assumption that V-shaped fingertips increase grasping robustness.

IV. DISCUSSION AND CONCLUSION

Experimental results show that our approach successfully provides an improved level of automation for adaptive finger production and usage. This level of automation is a step further towards fully automatic small batch size production since it potentially reduces the adaption time of assembly lines to new products and therefore reduces the time the assembly line is offline and not used.

This increased automation can also lead to improved manipulation flexibility and performance, as fingertips can be automatically changed during assembly operations to better match target objects. Traditionally, an assembly robot would need to use one universal finger-pair for all target objects.

These features also build the base for a future automatic fingertip design, production and application pipeline, which would reduce finger development and production time as well as decouple the design quality from the individual designer’s skills and experience. As previously stated, the human designer in the presented framework substitutes the automatic fingertip design module of such a future pipeline. This substitution enables us to prove and evaluate our framework’s basic concept and functionality before advancing our approach to such a fully automatic fingertip scheme.

Accordingly, future work will focus on further improving and extending the current production and task execution setup. As previously discussed, a fully automatic fingertip designer will be developed and interfaced with the production unit in order to achieve a fully automatic identification of the fingertips, required for a given manipulation scenario. Instead of rigid fingertips, future work could print flexible fingertips for certain use cases where a larger shape tolerance is required or the object is particularly fragile and (or) sensitive. This would also open the opportunity for a hybrid setup, where pre-produced universal fingers (rigid and soft) are stored in a standard library and get supplemented by customized fingers if the changed task scenario requires this. Accordingly, a potentially large amount of components could be handled by the standard fingers, while special and (or) new components can be manipulated with customized fingers automatically adapted and produced on the fly. This
would further reduce the adaption time of an assembly line to a new product.

ACKNOWLEDGMENT

The authors also would like to thank Robin Kirschner for her fruitful discussions and contributions.

REFERENCES

[1] H. Ha, S. Agrawal, and S. Song, “Fit2form: 3D generative model for robot gripper form design,” in Proc. Conf. Robot Learn., 2021, pp. 176–187.
[2] R. G. Brown and R. C. Brost, “A 3-D modular gripper design tool,” IEEE Trans. Robot. Automat., vol. 15, no. 1, pp. 174–186, Feb. 1999.
[3] P. Pedrazzoli, R. Rinaldi, and C. R. Boer, “A rule based approach to the gripper selection issue for the assembly process,” in Proc. IEEE Int. Symp. Assem. Task Plan. Assem. Disassembly 21st Century, 2001, pp. 202–207.
[4] D. T. Pham and E. Taccin, “An expert system for selection of robot grippers,” Expert Syst. With Appl., vol. 5, no. 3/4, pp. 289–300, 1992.
[5] D. T. Pham, N. S. Gourashi, and E. E. Eldukhri, “Automated configuration of gripper systems for assembly tasks,” Proc. Inst. Mech. Engineers, Part B: J. Eng. Manufacture, vol. 221, pp. 1643–1649, 2007.
[6] R. Saravanan, S. Ramabalan, N. G. R. Ebenezer, and C. Dharmaraja, “Evolutionary multi criteria design optimization of robot grippers,” Appl. Soft Comput. J., vol. 9, no. 1, pp. 159–172, 2009.
[7] F. Chen et al., “Topology optimized design, fabrication, and characterization of a soft cable-driven gripper,” IEEE Robot. Automat. Lett., vol. 3, no. 3, pp. 2463–2470, Jul. 2018.
[8] C.-H. Liu, T.-L. Chen, T.-Y. Pai, C.-H. Chiu, W.-G. Peng, and M.-C. Hsu, “Topology synthesis, prototype, and test of an industrial robot gripper with 3D printed compliant fingers for handling of fragile objects,” in Proc. IEEE WRC Symp. Adv. Robot. Automat., 2018, pp. 189–194.
[9] C.-H. Liu, F.-M. Chung, Y. Chen, C.-H. Chiu, and T.-L. Chen, “Optimal design of a motor-driven three-finger soft robotic gripper,” IEEE/ASME Trans. Mechatron., vol. 25, no. 4, pp. 1830–1840, Aug. 2020.
[10] M. Honarpardaz, M. Tarkian, J. Ölvander, and X. Feng, “Finger design automation for industrial robot grippers: A review,” Robot. Auton. Syst., vol. 87, pp. 104–119, 2017.
[11] M. Honarpardaz, M. Meier, and R. Haschke, “Fast grasp tool design: From force to form closure,” in Proc. IEEE 13th Conf. Automat. Sci. Eng., 2017, pp. 782–788.
[12] M. Honarpardaz, M. Tarkian, J. Ölvander, and X. Feng, “Experimental verification of design automation methods for robotic finger,” Robot. Auton. Syst., vol. 94, pp. 89–101, 2017.
[13] M. Honarpardaz, J. Ölvander, and M. Tarkian, “Fast finger design automation for industrial robots,” Robot. Auton. Syst., vol. 113, pp. 120–139, 2019.
[14] H. Song, M. Y. Wang, and K. Hang, “Finger tip surface optimization for robust grasping on contact primitives,” IEEE Robot. Automat. Lett., vol. 3 no. 2, pp. 742–749, Apr. 2018.
[15] L. Balan and G. M. Bone, “Automated gripper jaw design and grasp planning for sets of 3D objects,” J. Robot. Syst., vol. 20, pp. 147–162, 2003.
[16] V. B. Velasco and W. S. Newman, “Computer-assisted gripper and fixture customization using rapid-prototyping technology,” in Proc. IEEE Int. Conf. Robot. Automat., vol. 4, pp. 3658–3664, 1998.
[17] M. Kodnongbua, I. Good, Y. Lou, J. Lipton, and A. Schulz, “Computational design of passive grippers,” ACM Trans. Graph., vol. 41, no. 4, pp. 2–12, 2022.
[18] M. Nagel, F. Giese, and R. Becker, “Flexible gripper design through additive manufacturing,” in Proc. Robot., Fab. Architecture, Art Design, 2016, pp. 455–459.
[19] “Gripper finger configuration FGR,” Accessed: Nov. 24, 2022. [Online]. Available: https://schunk.de/de/de/fortraegunger-fgr
[20] B. S. Hornberg, R. K. Katzschmann, M. R. Dogar, and D. Rus, “Haptic identification of objects using a modular soft robotic gripper,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2015, pp. 1698–1705.
[21] P. J. Leitão et al., “Development and simulation of an automatic tool changer for an ABB robot” Master’s thesis, FEUP - Fac. Eng. Univ. Porto, 2019.
[22] B.-S. Ryuh, S. M. Park, and G. R. Pennock, “An automatic tool changer and integrated software for a robotic die polishing station,” Mechanism Mach. Theory, vol. 41, no. 4, pp. 4–432, 2006.
[23] C. Clévy, A. Hubert, J. Agnus, and N. Chaillet, “A micromanipulation cell including a tool changer,” J. Micromechanics Microeng., vol. 15, no. 10, 2005, Art. no. S292.
[24] T. Chagouri, F. Al-Darwish, A. Sharif, and Y. Al-Hamidi, “Product design journey: Novel tool changer,” in Proc. ASME Int. Mech. Eng. Congr. Expo., 2021, Art. no. V009T09A025.
[25] H. Ambrosio and M. Karamanoglu, “Design and development of an automatic tool changer for an articulated robot arm,” in Proc. IOP Conf. Series: Mater. Sci. Eng., 2014, Art. no. 012023.
[26] K. Brillkowski, “Sturdy tool changer,” Ind. Robot: An Int. J., vol. 23, 1996, pp. 25–28.
[27] P. Dhakal, “Design of automatic tool changer for universal collaborative robot,” Bachelor’s thesis, HAKM - HAKMTech Robot. Res. Group, Håme Univ. Appl. Sci., 2019.
[28] M. Babincsichi, “Automatic tool changer on collaborative robots,” Ph.D. dissertation, Universidade de Coimbra, Coimbra, Portugal, 2018.
[29] A. Pettinger, C. Dimoush, and M. Pyyry, “Passive tool changer development for an elastic and compliant manipulator,” in Proc. IEEE 15th Int. Conf. Automat. Sci. Eng., 2019, pp. 1200–1205.
[30] R. Berenstein, A. Wallach, P. E. Moudio, P. Cuellar, and K. Goldberg, “An open-access passive modular tool changing system for mobile manipulation robots,” in Proc. IEEE 14th Int. Conf. Automat. Sci. Eng., 2018, pp. 592–598.
[31] D. Gysinmoth and A. Toth, “Experimental evaluation of a novel automatic service robot tool changer,” in Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron., 2011, pp. 1046–1051.
[32] D. C. W. Friedman, J. Dosher, T. Kowalewski, J. Rosen, and B. Hannaford, “Automated tool handling for the trauma pod surgical robot,” in Proc. IEEE Int. Conf. Robot. Automat., 2007, pp. 1936–1941.
[33] S. McKinley et al., “An interchangeable surgical instrument system with application to supervised automation of multilateral tumor resection,” in Proc. IEEE Int. Conf. Automat. Sci. Eng., 2016, pp. 821–826.
[34] S. B. Backus and A. M. Dollar, “An adaptive three-fingered prismatic gripper with passive rotational joints,” IEEE Robot. Automat. Lett., vol. 1, no. 2, pp. 668–675, Jul. 2016.
[35] Z. Ren, N. Kashiri, C. Zhou, and N. G. Tsagarakis, “Heri II: A robust and flexible robotic hand based on modular finger design and under actuation principles,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2018, pp. 1449–1455.
[36] P. So, J. Wittmann, P. Ruhkamp, A. Sarabakha, and S. Haddadin, “Towards remote robotic competitions: An internet-connected task board and dashboard,” 2022, arXiv:2201.09565.
[37] G. Causey, “Guidelines for the design of robotic gripping systems,” Assembly Automat., vol. 23, 2003, pp. 18–28.
[38] “O.P.D. printers directly from Josef Prusa,” 2009. [Online]. Available: https://www.prusa3d.com
[39] J. M. Hiro Hronkó and J. Makarius, “OctoRest,” Jul. 10, 2016. [Online]. Available: https://github.com/dougbrison/OctoRest
[40] G. Häußge, “Octoprint,” Feb. 08, 2023. [Online]. Available: https://octoprint.org/
[41] L. Johannesmeier, M. Gerchow, and S. Haddadin, “A framework for robot manipulation: Skill formalism, meta learning and adaptive control,” in Proc. IEEE Int. Conf. Robot. Automat., 2019, pp. 5844–5850.
[42] F. Voigt, L. Johannesmeier, and S. Haddadin, “Multi-level structure vs. end-to-end-learning in high-performance tactile robotic manipulation,” in Proc. Conf. Robot Learn., 2021, pp. 2306–2316.
[43] J. Ringwald et al., “Printed fingertip geometry data table,” 2023. [Online]. Available: https://mediatum.ub.tum.de/1695927