2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

# Generalized Partially Destructive Disassembly Planning for Robotic Disassembly

Malte Hansjosten[1], Jan Baumgärtner[1] and Jürgen Fleischer[1]

*Abstract*— **While robotic assembly is a well researched topic, recycling and disassembly of products are also becoming ever more important as we transition to a more sustainable economy. In disassembly, we are typically only interested in a subset of product parts, which opens the possibility of using destructive processes such as tearing, cutting, or milling to speed up the disassembly. Currently, such destructive actions are only included as predefined case-specific actions such as milling away a screw head. By contrast, this paper presents a generalized approach to destructive disassembly planning that can automatically derive destructive disassembly actions from a symbolic representation of the disassembly state. Viable destructive actions are identified and verified only based on the underlying geometric model, circumventing the need for their explicit definition. We showcase the performance of this system both virtually on several test parts and physically by destructively and non-destructively disassembling a model of an electric motor using a robot manipulator with a multitool end effector.**

## I. INTRODUCTION

To mitigate the unsustainable use of Earth's limited resources, a shift towards a circular economy is essential, where used products are recirculated into the economy rather than discarded. An economical implementation of such approaches requires new solutions, especially for the automated disassembly of products. In contrast to automated assembly systems, disassembly systems need to be more flexible to deal with products of varying and often unknown conditions. This makes robots coupled with interchangeable tools a natural fit for this task as they can be easily reprogrammed for different disassembly targets. However, to effectively carry out disassembly tasks, a robot needs to be able to plan its actions and motions based on the product's specifications. In the current robotics research, this behavior is often learned using reinforcement learning. Examples of this are the work of [1] or [2]. However, looking at typical benchmarks in this area such as [3] in which IKEA furniture is assembled, we see that the assembly tasks involve less than 20 parts. But more complex products such as smartphones or car parts typically have much more parts in more complex relations. For such products, assembly steps are typically determined by classical planning algorithms. This is typically done by converting the CAD model into a graph representation and then using graph search algorithms to find an assembly sequence [4]. The assembly actions are thus represented symbolically and only later translated into part motions. Often these systems also start with an assembled product, try to disassemble it, and then invert the sequence to get an assembly sequence [5]. Thus most assembly planners are also disassembly planners. However, this does not recognize the fact that disassembly can also use destructive processes that are not reversible and therefore outside the scope of the mentioned (dis-)assembly planners. A great overview of the field of disassembly planning is given by [6], which also distinguishes between destructive and non-destructive disassembly. Destructive disassembly recognizes that some parts of a product are not intended to be reused and it thus might be more efficient to destroy them. Even so, works in this area typically treat destruction as a backup that is only used if nondestructive assembly is not possible [7], or when traditional disassembly fails [8].

We want to argue that destructive disassembly should be considered as a viable alternative to non-destructive methods, offering potential simplifications and accelerations in the disassembly process. Locking rings for example can be disassembled non-destructively in principle, however automating this process is very challenging, due to limited accessibility and the need for significant force application. Considering the minimal material value of such components, destructive disassembly could be a viable alternative. It can also reduce the hardware complexity of automated systems. One milling tool could disassemble different screws as well as rivets, which can not be disassembled non-destructively. This also opens up completely new sequences, depending on the disassembly target. In electric motors, for example, the rare earth minerals used in permanent magnets are a highly valuable resource, with a critical supply chain [9].

Traditional recycling approaches usually include shredding the motor followed by different sorting processes to recover the used material. In such cases, magnet material can hardly be separated from other materials, making disassembly desirable [10]. While other used materials, like copper or aluminum, can also benefit from disassembly and subsequently higher purity of the recycled material, their recovery from material mixtures is much easier. This means that an overall optimal disassembly sequence might involve quickly but destructively disassembling a motor housing to get to the rotor and magnets. These can then be reused, while the destroyed housing is fed to established material recycling processes.

While the examples above showcase the possible uses of destructive disassembly as an equal alternative to nondestruc-

*The authors gratefully acknowledge financial funding from the German Federal Ministry of Economic Affairs and Climate Action (grant no. 13IK003H) as well as organizational support by the VDI Technologiezentrum GmbH.

[1]Malte Hansjosten, Jan Baumgärtner, and Jürgen Fleischer are with the with wbk Institute of Production Science, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany malte.hansjosten@kit.edu

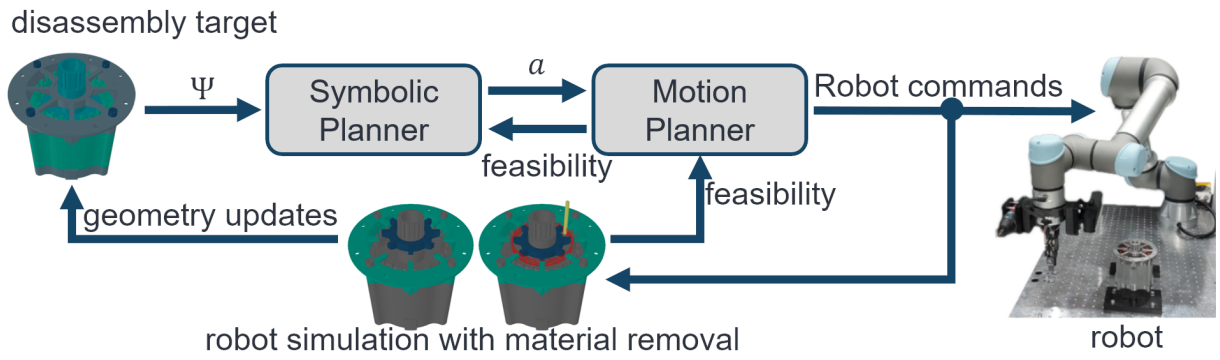



Fig. 1: System overview. The symbolic planner manipulates the symbolic state $\Psi$ to generate a sequence of actions $a$. The motion planner then translates these actions into robot motions. The feasibility of each action is checked by a simulation which also includes the material removal process.

tive processes, new planning approaches for such instances are needed. The flexibility of destructive disassembly is also the reason why a generalized planning algorithm for destructive disassembly is a difficult proposition. Since there are infinite ways to remove material from a component there are infinite possible destructive disassembly processes. While manually defining task-specific actions, like the removal of a screw head through milling, is of course a valid approach, this can not be generalized. We would like to further the development of generalized disassembly planners by proposing an integrated task and motion planner that can automatically derive both destructive and nondestructive disassembly sequences from a symbolic representation of an object's disassembly state. The main contributions are:

- A symbolic representation of the disassembly state that allows us to model destructive and non-destructive actions.
- A motion planning system that can plan motions for both destructive and non-destructive actions.

An overview of the system can be seen in Fig. 1.

## II. Related Work

There have already been several works that have looked at destructive disassembly. Chiefly among them is the work of [11] in which the authors present a motion planning system for the destructive disassembly of electronic components. However, as in previous works the authors treat the destructive disassembly like a task-specific action which in this case mills away screws. It is thus incapable of planning more flexible or complex destructive actions. Our work can thus be seen as a generalization using a skill-based approach to disassembly planning. The general architecture follows a hierarchical framework as described in [12] with a symbolic planner manipulating an abstract description of the disassembly state, which is then translated into robot motions using a set of skills. We extend this approach using the iterative feasibility checking framework described in [13] but extended to deal with destructive actions where collision shapes change over time. This was achieved using the pybullet industrial framework [14] which allows us to simulate material removal along with robot motions.

## III. Methodology

### A. Symbolic Planner

*1) Geometric motion constraints:* The symbolic representation is used by the high-level planner to determine which destructive or non-destructive disassembly action can be used. This necessitates a suitable representation of the current state of an assembly. This could be a simple graph where nodes represent parts and edges represent connections between those parts. However, this would only allow us to represent destructive actions that completely remove a connection or part, which is not realistic since many destructive actions only change how the parts can move relative to each other. A simple example can be seen in Fig. 2, where a rivet joint is dissolved by milling away part of the rivet's head, changing the relation between the rivet and plate A. All parts are still connected but plate A can now be pulled away from plate B.

For this reason, we need to model the relation between the two parts in terms of the movement possible between them. In this case, we only look at infinitesimal movements and thus model the relation between two parts $i$ and $j$ as a set of movement directions. This can be expressed as a binary function $\psi_{ij}(\theta,\phi)$ of the following form:

$$\psi_{ij}(\theta,\phi) = \begin{cases} 1 & \text{if movement in direction is possible} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $\theta$ and $\phi$ are the spherical coordinates of the movement direction. Note that for two given parts there are two such functions $\psi_{ij}$ and $\psi_{ji}$. However these follow the relation $\psi_{ij}(\theta,\phi) = \bar{\psi}_{ji}(\theta,\phi)$. Computing this function for all pairs of parts gives us a matrix $\Psi$, which will serve as the symbolic representation of the disassembly state. This matrix can be seen as an extension of the adjacency matrix of a graph used in approaches such as [15] or [16].

Its elements $\psi_{ij}$ can be obtained by determining contact points and the corresponding contact normals between the

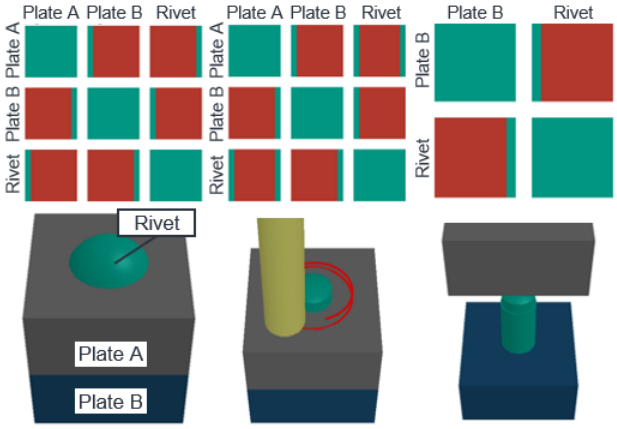



Fig. 2: Milling removes part of a rivet's head, enabling movement for plate A previously blocked by the rivet, as shown in the action sequence (left) and matrices $\Psi$ (right). Red areas indicate blocked directions $\theta, \phi$, while green signifies possible movements. The first column of $\Psi$ reveals the new global movement direction through a green channel.

two parts. To compensate for inaccuracies resulting from the triangulation of the 3D-mesh geometries we are working with and also simplify the comparison and identification of different assembly states the values of $\theta$ and $\phi$ are discretized. Each contact normal is then binned into such a discrete direction and all $\psi_{ij}(\theta,\phi)$ on the hemisphere around that direction are set to 0. A graphical representation of this matrix can be seen on the right in Fig. 2, where each element is a surface plot of the function $\psi_{ij}$. Based on the symbolic state $\Psi$ we can now define a set of actions that can be used to change the state. Here we differentiate between two types of actions, destructive and non-destructive actions. Non-destructive actions are actions that remove a part from the assembly without destroying it. As such their result is the removal of a column and row from the matrix $\Psi$. However, they require there to be at least one direction in which the movement of the part is not constrained by any other part. The prerequisite for a non-destructive action $a(i)$ can thus be expressed as:

$$preq(a(i)) = \begin{cases} 1 & \exists(\theta,\phi) : \prod_j \psi_{ij}(\theta,\phi) = 1 \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Destructive disassembly actions on the other hand typically don't completely remove a part but instead change its geometry in such a way that new movement directions become possible. This means that they effectively change the value of the elements $\psi_{ij}$ and $\psi_{ji}$ from 0 to 1 for at least one vector $(\theta,\phi)$ and one part pair ij. However, the changes of $\psi_{ij}$ have to be made such that they can be executed through material removal. Although it would in theory be possible to determine the material that needs to be removed to change $\psi_{ij}$ from 0 to 1 for any $(\theta,\phi)$ this would lead to an unreasonably high number of possibilities. Our approach aims to identify specific finite destructive assembly possibilities that can be integrated into the planning process. Currently, we focus on cases where removing material directly affects the relationship between two parts, enabling the disassembly of one of them. While our approach may thus not identify all destructive disassembly options, the example cases in section IV demonstrate its validity for a substantial range of scenarios. Typically, parts are assembled along a primary axis defined by their geometry and secured in place by a secondary geometry. This secondary geometry can be created by joint components (like screws or rivets) or even by the parts themselves (e.g., a lid fitting around a bearing). Examples can be found in Fig. 5, where plates are held by rivets, rings are secured by housing, and a bearing is kept on a shaft with a locking ring. To enable disassembly, we must identify the geometry blocking the part's movement along its primary axis of movement.

To illustrate this approach, let's consider the rivet connection in Fig. 2. Specifically, we'll focus on the movement of plate A relative to the rivet. In this case, we observe that plate A can only move downward, while the rivet can only move upward, making the z-axis the primary axis of movement for this pair. However, both directions are restricted by the bottom plate connected to the rivet. This type of joint cannot be disassembled non-destructively. Yet, with destructive disassembly, we can remove plate A by milling the portion of the rivet's head that constrains plate A along the positive z-axis. This manipulation alters $\psi_{ij}$ and $\psi_{ji}$, enabling reverse movement of plate A relative to the rivet. This idea will be the cornerstone of the simplified symbolic destructive action model. Given a main direction of movement $v$ between two parts, which can be computed as:

$$v = \frac{1}{n} \sum_{\theta,\phi} \psi_{ij}(\theta,\phi) \quad (3)$$

where $n$ is the number of possible movement directions.

We can then define a destructive action $a(i,j)$ as an action that allows for the reverse movement of $v$ of part i relative to part j. Since the removal of material might affect other elements of $\psi_{ij}$ and $\psi_{ji}$ it is necessary to compute the exact geometry that is removed and recompute the elements of $\psi_{ij}$ and $\psi_{ji}$ associated with the changed part. The change in the associated elements $\psi_{ij}$ and $\psi_{ji}$ for an action $a(i,j)$ can be computed by projecting the 3D-mesh model of the part that is not destroyed along the movement direction $-v$.

In practice, for each vertex in the 3D-mesh a ray is projected along $-v$, and intersections of the ray with the part to be changed destructively are determined. In a first approximation, we then calculate a minimum volume cylinder containing all intersection points, and a maximum volume cylinder with the same axis as the first cylinder, that does not contain any of the intersection points (see Fig. 3). The volume between the cylinder hulls effectively computes the material that needs to be removed to allow for the reverse movement of $v$. This results in a new geometry whose contact points with other parts can be used to compute the new values for $\psi_{ij}$ and $\psi_{ji}$. This is again illustrated in Fig. 2 where part of the head of the rivet is removed as a result of this

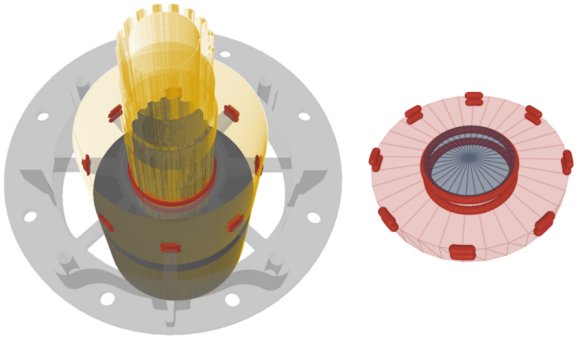


Fig. 3: Computation of geometry to be removed. Left: Projected rays (yellow) and intersections (red). Right: Minimum volume cylinder containing all intersection points (red) and maximum volume cylinder not containing any intersection points.

calculation.

*2) Non geometric motion constraints:* While the symbolic state $\Psi$ is a good representation of the geometric relationship between parts, it might not capture all constraints that are relevant for disassembly. For example, some parts might be connected by screw threads, glue, or other methods that are not modeled geometrically. These will require special actions and tools to remove them. This can be achieved by extending the symbolic state $\Psi$ with additional binary constraint matrices $\tilde{\Psi}_k$ where $k$ is a relation between two parts that requires a special action. For example $\tilde{\Psi}_{screw}$ could be a matrix that encodes all parts that are connected by screw threads and thus need a screwdriver to be removed. The prerequisite for a non-destructive action $a(i,k)$ can then be extended to:

$$preq(a(i,k)) = \begin{cases} 1 & \exists(\theta,\phi) : \prod_{!k} \prod_j \psi_{ij}(\theta,\phi)\tilde{\psi}_{ij_k} = 1 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

In this case $\prod_{!k}$ encodes an elementwise multiplication over all tool-specific constraints $\tilde{Psi}_k$ except those associated with relation $k$.

### B. Motion Planner

The motion planner translates symbolic $\Psi$ manipulation into executable robot motions. While we can always perform a destructive action symbolically, this might not be possible in practice. Here we first have to check that the tool in particular and the robot in general can reach the part. The same is also true for non-destructive actions, here we might not only be incapable of gripping a part, but there might also be other parts in the way that prevent us from reaching the part. Unfortunately, this can not be checked symbolically since reachability depends on the toolpath and thus on the motion planning. Each action type has therefore a feasibility check incorporated into its associated motion planning algorithm. In general, these actions are movements and tool-specific actions like opening or closing a gripper or (de-)activating a screwdriver. Planning a disassembly action correspondingly involves determining specific points or poses relative to the part to be disassembled, planning collision-free movements to transition between those points, and parameterization of tool-specific actions. These steps can be planned individually using established frameworks wherever possible.

For general motion planning for example we opt to utilize motion planning frameworks such as [17], while grip points can be determined through grasp planning algorithms as exemplified in [18]. Planning a gripping action consists of determining a suitable grasp, calculating the necessary movement to reach this grip point, closing the gripper, and moving away in a direction the part can move in. The disassembly direction can be determined directly from the state $\Psi$, as the main axis of movement for the part concerning all other parts, not just relative to one other part, analogous to (2) and (3). Planning a screwing action initially necessitates motion planning to move the tip of the screwdriver to the screw head, align it with the screw axis, and move the activated tool along the screw axis. This axis is defined by $\Psi$ analogous to the disassembly direction of gripped parts. It's significant to mention that practical scenarios might require screw removal post-unscrewing by an additional gripping action. For this work, the corresponding gripping action is seen as part of the screwing action.

In each case, the feasibility check of the action is then a reachability check for both the tool and robot arm. It should be noted that the real part might deviate from the CAD model used for this check which is especially common after years of use or in the case of wear and tear or destructive disassembly. Future work will therefore have to investigate the use of sensor data to adapt the motion plan to the real part geometry.

Contrary to non-destructive actions, robotic motion planning for destructive disassembly actions is largely unsolved, especially for complex geometries. The main challenge here is to identify a suitable toolpath that achieves the $\psi_{ij}$ changes defined by the symbolic planner (see section III-A). Our approach focuses on the application of a milling process, due to its flexibility in possible material removal. A simple solution, in this case, is to calculate helical paths, offset from the contour of the calculated volume to be destroyed by half the mill diameter, and then iteratively offset these curves by a full mill diameter from each other until all material is removed. Another possible approach would be to implement a slicing tool commonly used in 3D printing for a *layered* removal of material. However, the calculation of helical paths opens up the possibility of significantly simplifying the destructive disassembly process. Depending on the specific situation it is possible that the desired change to $\psi_{ij}$ can also be achieved by a combination of milling and gripping action, analogous to the subsequent removal of a screw through gripping. Milling a helical path around the outside (or inside) of the calculated volume to be destroyed might result in a new, disjunct part that is no longer connected to the remainder of the original part. This new part can be removed by a simple gripping action, thus significantly reducing the necessary milling time and the amount of created chips. An example of this is depicted in Fig. 4, where only part of the upper lid has to be destroyed to get to the inside component

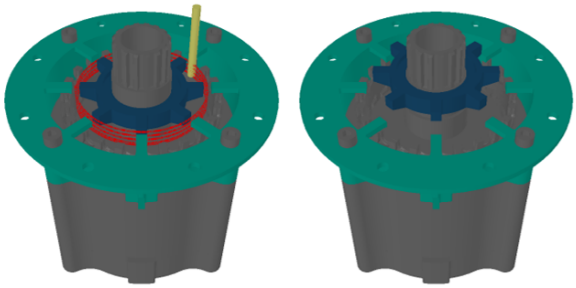


Fig. 4: Destructive disassembly of simple dummy motor by milling and subsequent gripping.

of the depicted electric motor. To check the feasibility of a milling process, we have to check whether each point on the toolpath is reachable by both tool and robot arm. Since milling necessitates the collision of the milling tool with the material to be removed such collisions are filtered from the reachability check. It has to be noted that process parameters, like cutting speed or cutting depth, are currently static values which have to be chosen very conservatively to avoide process failures. In the future these could be set dinamically, for example based on part material, to maximize material removal.

## IV. Experiments

To test the overall planning approach several simulative and physical experiments have been conducted. Firstly two test cases were defined, that represent an abstract generalization of possible ways to join multiple parts together. For parts to be held together geometrically, they require a surrounding structure. This structure can be created either by a third part that passes through the others (e.g. a screw or rivet) or encircles them (e.g. a housing or clamp).

The first two test cases shown in Fig. 5 represent these two options. The first is the same rivet connection, already shown in Fig. 2. The second test case consists of two ring-shaped parts inside a closed housing (see Fig. 5 middle). Additionally, a third test case was defined, which consists of a bearing that is secured on a shaft by a locking ring (see Fig. 5 right). This test case can formally be seen as equal to the first test case since the physical frame around the bearing is established by the shaft and the shaft shoulder on one side of the bearing and the locking ring on the other side. However, it serves as an example where destructive disassembly can automate a process that is hard to automate nondestructively.

To test whether these simple examples extend to more complicated parts, two additional cases were defined using electric motors as examples. One is a simple dummy motor (see Fig. 4) consisting of two lids, a stator, a rotor, and four screws. The other is a more complex, realistic motor assembly consisting of 14 parts, including different screws and a locking ring. For all examples the planner was tasked with finding different possible disassembly sequences for which

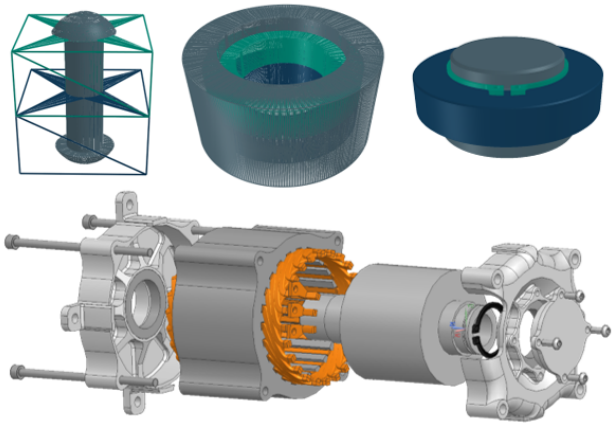


Fig. 5: Test cases for validation experiments. Left: Rivet joint. Middle: Parts in housing. Right: Locking ring. Bottom: Complex motor assembly.

virtual feasibility checks were performed using pybullet industrial [14]. All calculations were executed on a Lenovo Thinkstation P360 with an Intel® Core™ I9-12900K processor. An overview of the planning results including necessary computation times can be seen in Table I. Additionally, a robotic setup was commissioned which consists of a robot arm and a multitool end effector for screwing, gripping, and milling (see Fig. 1 right). Our proposed planning system was tasked to try to disassemble the simple dummy motor twice, once allowing destructive actions and once forbidding them.

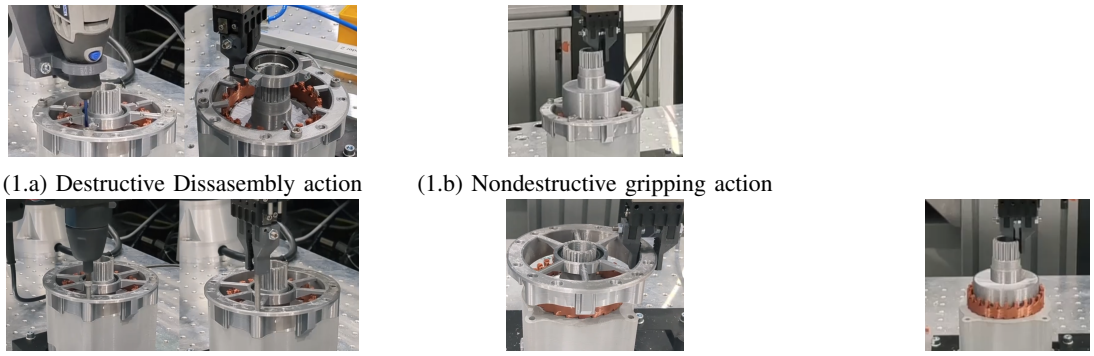
## V. Results

Through the virtual disassembly experiments, we were able to show that our approach can generate valid disassembly sequences for all defined test cases without the need to manually define the disassembly operations. Plotting the resulting state matrices $\Psi$ one can observe how they are changed through the application of destructive disassembly processes. The results for the first test case (rivet joint) can be seen in Fig. 2, and the corresponding results for the second and third test cases can be seen in Fig. 7.

In each case the destructive process adds a new possible movement direction to one of the parts, allowing their disassembly by a simple gripping action. A similar simplification can be seen in the dummy motor disassembly. As indicated by the keyframes in Fig. 6 both destructive and non-destructive disassembly sequences were successfully executed. However, the destructive disassembly sequence

TABLE I: Planning results and computation times for different test cases.

| Assembly | Target | Sequences | Endstates | Time |
|---|---|---|---|---|
| Rivet | All | 8 | 1 | 17.4s |
| Parts in housing | All | 8 | 1 | 17.0s |
| Locking ring | All | 2 | 1 | 13s |
| Simple dummy motor | Rotor | 50 | 4 | 139.1s |
| Complex motor | Rotor | 11688 | 4 | 1683.0s |



(1.a) Destructive Dissasembly action (1.b) Nondestructive gripping action

(2.a) Nondestructive screwing action (2.b) Nondestructive gripping action (2.c) Nondestructive gripping action

Fig. 6: Key frames from robotic disassembly operations. Top: destructive. Bottom: Non-destructive.

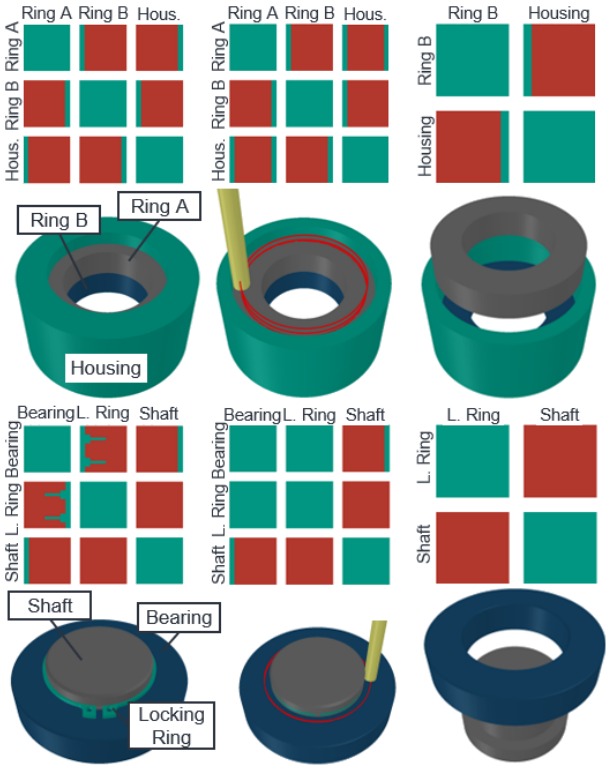



Fig. 7: Action sequence (bottom) and associated matrices $\Psi$ (top) planned for test cases two and three. Red areas indicate blocked directions $\theta, \phi$, while green signifies possible movements.

was significantly simpler forgoing the need for a screwdriver and removal of the full lid.

## VI. Conclusion

In this paper, we presented our contribution to the development of more generalized disassembly planning systems that incorporate both destructive and non-destructive processes. The basis for our approach is a symbolic representation of an assembly state that models relations between part pairs as binary functions for possible movement directions. Based on this, the disassembly planner we describe in this work can identify viable disassembly processes and sequences, both destructive and non-destructive. One improvement over other planning systems is that destructive disassembly processes do not have to be specifically defined but are derived automatically from the part geometry. The underlying motion planner translates the symbolic plan into movements and tool-specific actions that can be carried out by a robot system.

We tested the system both in simulations and on a robotic demonstrator and were able to successfully implement both destructive and non-destructive disassembly sequences generated by the planner. While this showcases the possibilities of incorporating both destructive and non-destructive processes in a generalized way into disassembly planning, several limitations have to be acknowledged and many points are left for future investigation. For one, our planner only includes disassembly processes that directly allow at least one part to be removed from the assembly. For example, cases, where material removal is an intermediary step to reach a part that is not in direct contact with the destroyed part are currently not considered. Also, since the aim of our approach is not to find all possibilities for destructive assembly, but to identify a subset of generally viable possibilities, there might be more optimal alternatives that are currently overlooked. This is exacerbated by the approximation of the removed material through cylinders. This might lead to unnecessary material removal or cases were the planner fails to find solutions due to this simplification. Further, the decision on which sequence is chosen depends on the costs associated with the individual processes. Herein lies much room for investigation and optimization to determine ideal cost functions that best model real circumstances. In addition, the practicality of the generated sequences and actions has to be investigated. Especially the influence of the destructive processes on the rest of the assembly (i.e. through the resulting debris) has to be considered. Results could then be included in the aforementioned cost functions. So while the automated and generalized planning of destructive disassembly as an equal alternative to non-destructive disassembly opens up many new possibilities, the approach presented in this work is only a first step and the topic should be investigated further.

## REFERENCES


[1] L. Ma, J. Gong, H. Xu, H. Chen, H. Zhao, W. Huang, and G. Zhou, "Planning assembly sequence with graph transformer," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 12 395–12 401.

[2] Y. Lin, A. S. Wang, E. Undersander, and A. Rai, "Efficient and interpretable robot manipulation with graph neural networks," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2740–2747, 2022.

[3] Y. Lee, E. S. Hu, and J. J. Lim, "Ikea furniture assembly environment for long-horizon complex manipulation tasks," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 6343–6349.

[4] S. Münker and R. H. Schmitt, "Cad-based and/or graph generation algorithms in (dis)assembly sequence planning of complex products," *Procedia CIRP*, vol. 106, pp. 144–149, 2022, 9th CIRP Conference on Assembly Technology and Systems. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827122001706

[5] D. Halperin, J.-C. Latombe, and R. H. Wilson, "A general framework for assembly planning: The motion space approach," *Algorithmica*, vol. 26, no. 3, pp. 577–601, Apr 2000. [Online]. Available: https://doi.org/10.1007/s004539910025

[6] S. K. Ong, M. M. L. Chang, and A. Y. C. Nee, "Product disassembly sequence planning: state-of-the-art, challenges, opportunities and future directions," *International Journal of Production Research*, vol. 59, no. 11, pp. 3493–3508, 2021. [Online]. Available: https://doi.org/10.1080/00207543.2020.1868598

[7] X. Song, W. Zhou, X. Pan, and K. Feng, "Disassembly sequence planning for electro-mechanical products under a partial destructive mode," *Assembly Automation*, vol. 34, no. 1, pp. 106–114, Jan 2014. [Online]. Available: https://doi.org/10.1108/AA-01-2013-006

[8] Y. Laili, X. Li, Y. Wang, L. Ren, and X. Wang, "Robotic disassembly sequence planning with backup actions," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 2095–2107, 2022.

[9] T. Elwert, D. Goldmann, F. Römer, M. Buchert, C. Merz, D. Schueler, and J. Sutter, "Current developments and challenges in the recycling of key components of (hybrid) electric vehicles," *Recycling*, vol. 1, no. 1, pp. 25–60, 2016. [Online]. Available: https://www.mdpi.com/2313-4321/1/1/25

[10] T. Elwert, D. Goldmann, F. Roemer, and S. Schwarz, "Recycling of ndfeb magnets from electric drive motors of (hybrid) electric vehicles," *Journal of Sustainable Metallurgy*, vol. 3, no. 1, pp. 108–121, Mar 2017. [Online]. Available: https://doi.org/10.1007/s40831-016-0085-1

[11] W. H. Chen, G. Foo, S. Kara, and M. Pagnucco, "Automated generation and execution of disassembly actions," *Robotics and Computer-Integrated Manufacturing*, vol. 68, p. 102056, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584520302672

[12] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1470–1477.

[13] I. Rodrıguez, K. Nottensteiner, D. Leidner, M. Kaßecker, F. Stulp, and A. Albu-Schäffer, "Iteratively refined feasibility checks in robotic assembly sequence planning," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1416–1423, 2019.

[14] J. Baumgärtner, M. Hansjosten, D. Schönhofen, and P. D.-I. J. Fleischer, "Pybullet industrial: A process-aware robot simulation," *Journal of Open Source Software*, vol. 8, no. 85, p. 5174, 2023. [Online]. Available: https://doi.org/10.21105/joss.05174

[15] R. Viganò and G. Osorio Gómez, "Assembly planning with automated retrieval of assembly sequences from cad model information," *Assembly Automation*, vol. 32, no. 4, pp. 347–360, Jan 2012. [Online]. Available: https://doi.org/10.1108/01445151211262410

[16] B. Imen, H. Moncef, T. Moez, and A. Nizar, "Generation of disassembly plans and quality assessment based on cad data," *International Journal of Computer Integrated Manufacturing*, vol. 33, no. 12, pp. 1300–1320, 2020. [Online]. Available: https://doi.org/10.1080/0951192X.2020.1815852

[17] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *2009 IEEE international conference on robotics and automation*. IEEE, 2009, pp. 489–494.

[18] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic grasp planning using shape primitives," in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 2. IEEE, 2003, pp. 1824–1829.