Gripping Point Determination and Collision Prevention in a Bin-Picking application

Dipl.-Inf. Felix Spenrath, Dipl.-Ing. Alexander Spiller, Univ.-Prof. Dr.-Ing. Dr. h. c. Alexander Verl
Fraunhofer Institute for Manufacturing Engineering and Automation IPA, Stuttgart, Germany

Abstract

Bin-Picking is a complex subject. Attempts of industrial realizations are often too slow or too unreliable. Many approaches strongly focus on detecting the pose of objects inside the bin. While this is an essential part of bin-picking, industrial realizations also need to have a robust strategy to pick the detected objects out of the bin, even in complex situations. We address this issue by separating the object pose detection from the task of finding an appropriate gripping position. Our goal is to find a suitable position for the gripper and avoid collisions with the bin or other objects. This collision-free and fast gripping point determination will be presented in this paper. The basic idea is to generate several potential gripping configurations and to rate them, primarily based on the probability of collisions with any obstacles inside the sensor point cloud. The implemented approach has produced good results in experiments and industrial applications.

1 Introduction

Most bin-picking applications focus on identifying and detecting the position of objects inside the bin. However, to successfully grip an object, it is necessary to find a suitable pose for the gripper at the moment of gripping as well as a path to approach this pose and a path to depart from it. Although simple objects may be gripped with a fixed gripper orientation, objects with limited grippable areas require a more complex approach.
In [1] it has already been shown that this can be done by testing for collisions of the gripper with both the box (which has a known size and position) and other detected objects. Collisions with other obstacles, like objects that have not been detected or a deformed part of the bin, cannot be prevented this way. To detect these, the sensor data needed to detect the objects, e. g. a point cloud, can be used to detect collisions. This could be achieved by creating a hull object structure representing the occupied areas inside the bin [2] and testing for collisions with this structure. Another approach is to test for collisions with the point cloud directly by determining which points are in the gripper model and how deep they are inside [3].
We present a fast and industrial-suited algorithm which creates several gripping solutions and tests each solution for collisions with the point cloud. Since one of our main goals is speed and it is important to detect close collisions as well, we do not determine the penetration depth of the points nor do we need to calculate if a point is inside or outside the gripper model.

1.1 Overview of our bin-picking application

Our bin-picking application is divided in two main modules. The first module is the Object Pose Detection (OPD) which identifies and locates objects. The module used for the experiments is an improved version of the OPD shown in [1]. The second module is the Gripping Point Calculation (GPC), which calculates a position for the gripper to grip the detected object as well as the path into and out of the bin. The GPC is subject of this paper.
The OPD is based on a point cloud generated by e. g. a laser scanner which is panned or translated above the bin, either by a separate actuator or by the robot itself. This point cloud is used by the GPC to detect potential collisions on possible gripping configurations and, if a collision is detected, to select another one.

2 Preparations

To calculate the best gripping solution the algorithm needs some information about the used grippers and objects to grip, including CAD models and definitions of grippers and possible gripping points on the objects. The necessary data and the frames used in the context of the Gripping Point Determination will be explained in this section.

2.1 CAD models

Since the algorithm relies on detecting collisions on possible gripping solutions, CAD models of the used grippers are needed. However, since this CAD model is used for a collision test, it should be simplified to increase performance. The process of simplifying the gripper model is currently done manually. The main goal of the simplification process is to reduce the number of triangles in the model while considering two contrary aspects. On one hand, to detect collisions with all parts of the gripper, no part of the original gripper model should be outside the simplified one. On the other hand, the simplified model should not be too large either to not prevent possible grips.
The algorithm also loads CAD models of the objects to grip. However, these are currently used only for the visualization of the gripping situation.

2.2 Frames

Before explaining the parameters that need to be configured, we introduce several frames that will be used to describe them. **Figure 1** visualizes these frames.

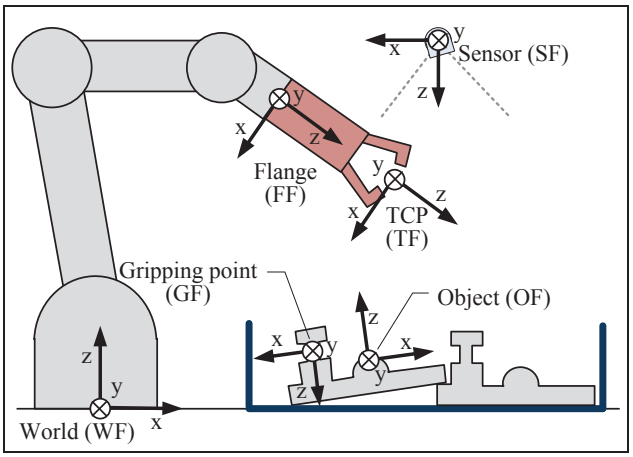

Figure 1 Frames used in the GPC.

This approach uses the following frames:

- world frame (**WF**): fixed to the base of the robot.
- sensor frame (**SF**): fixed to the initial pose of the sensor. The SF always stays the same relative to the WF, even when the sensor itself is panned or translated during the scan process.
- flange frame (**FF**): fixed to the robot flange.
- TCP frame (**TF**): fixed to the Tool Center Point (TCP) of the gripper.
- object frame (**OF**): fixed to the object (to the center of its bounding box in the CAD model).
- gripping point frame (**GF**): fixed to the gripping point.

Note that a gripper may have several TCPs and an object usually has several gripping points.

2.3 Grippers, TCPs and gripping points

In addition to the CAD models, grippers and work piece objects are defined in an XML file. Particularly two types of homogenous transformations need to be defined using the aforementioned frames:

- Transformations T_{FF}^{TF} which describe the position and rotation of the TCPs on the grippers relative to the flange frame.
- Transformations T_{OF}^{GF} which describe the position of gripping points on the objects relative to the object frame. These gripping points identify spots on an object it can be gripped on.

3 Algorithm overview

For each object, detected by the OPD, the algorithm generates several gripping solutions. These gripping solutions represent the real world position of the gripper at the gripping point. Then, for all gripping solutions (possibly limited by certain conditions) an approach and a depart path is generated. These paths consist of several gripper poses and usually connect the gripping point with a safe position above the bin. These poses are defined by using the different frames introduced before, particularly the world frame and the gripping point frame.

Now a rating is calculated for each gripping solution, primarily based on the risk of collisions. Finally, the best gripping solution is selected and the corresponding path poses are transmitted to the robot.

The individual parts of the algorithm will be described in more detail in the following sections.

4 Generation of gripping solutions

For each object pose (in the world frame) returned by the OPD, several gripping solutions are generated. Generally, each defined gripping point results in one gripping solution on each of the objects. So if, for example, two gripping points are defined and the OPD returns three objects, it would result in six gripping solutions being generated. To create a gripping solution, the gripping points, defined in the object frame, are transformed into the world frame using the detected position and rotation of the object described by the transformation T_{WF}^{OF}. This is done by using the formula:

$$T_{WF}^{GF} = T_{WF}^{OF} \cdot T_{OF}^{GF}$$

The TCPs are defined in such manner, that by aligning the TCF frame with the gripper frame, the gripper would be in the optimal gripping position. Therefore at the moment of gripping, it is:

$$T_{WF}^{TF} = T_{WF}^{GF}$$

Using the position of the TCP, one can now calculate the position and rotation of the flange in the WF:

$$T_{WF}^{FF} = T_{WF}^{TF} \cdot (T_{FF}^{TF})^{-1}$$

All these transformations at the moment of gripping are shown in a 2D example in **Figure 2**.

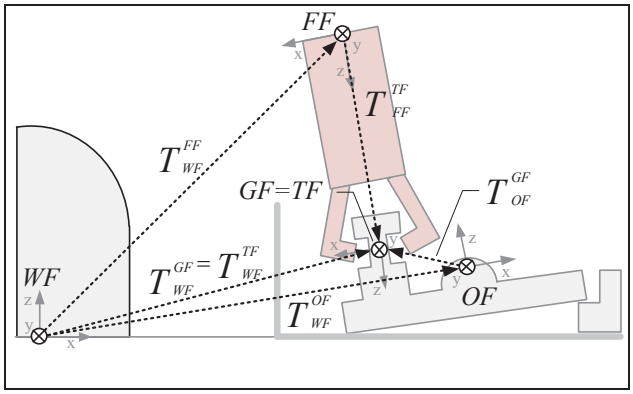

Figure 2 Transformations used for calculating a gripping solution and the flange position in the world frame.

The generated gripping solutions contain all these transformations, particularly the position and rotation of the gripper and the object in the world frame, and can therefore be visualized inside the point cloud (**Figure 3**).

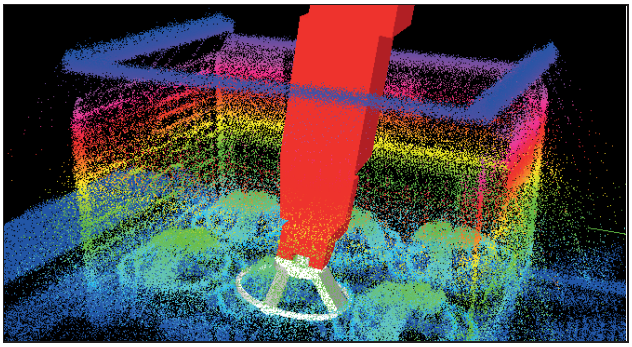

Figure 3 Visualization of a gripping solution. The gripper is displayed in red and the work piece object in white. The color of the point cloud represents the height.

4.1 Adaption of the predefined Gripping Points via Actions

Since the best suited position for the gripper may depend heavily on the position of the object, the predefined gripping points are not definite, but can be modified by so called actions. These actions are defined along with the gripping points and can change the position and orientation of the gripping point and therefor the gripper.
Figure 4 visualizes an appropriate scenario for the use of such an action. The used gripper has a large chassis part on one side, so it may be preferable to have this large part directed towards the center of the bin. This can be done by defining an appropriate action, provided of course, that such a rotation around the TCP does not render the grip impossible.

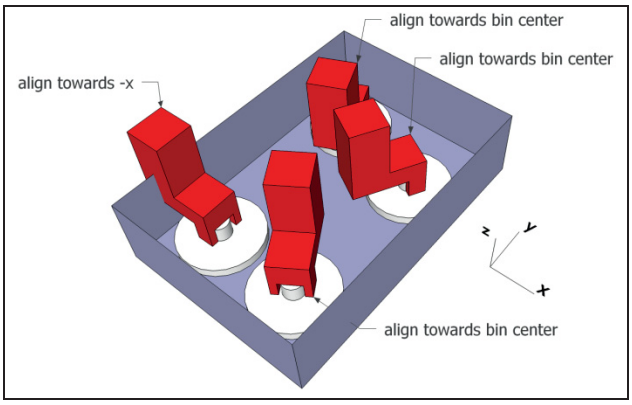

Figure 4 An appropriate scenario for the use of an action. The three gripping solutions on the right hand side were modified by an action aligning them towards the center of the bin. As a counterexample, the other gripping solution was modified by an action aligning it towards $-x$, which is more likely to cause a collision.

Another possible scenario is the translation of e. g. a magnetic gripper to move it as far away as possible from the borders of the bin.
Of course the possibility to use these actions depends heavily on the used gripper, because the performed transformations must always result in a possible grip.

4.2 Multiplication of Gripping Solutions

Another way to deal with the scenario in **Figure 4** is to generate several gripping solutions with different orientations, which are all processed and tested for collisions. This can be done by simply defining one gripping point and providing the information how the other gripping points should differ from the defined one (e. g. rotated around the z-axis of the TCP) and how many gripping solutions should be created from this one gripping point. **Figure 5** visualizes this situation. This allows the simple generation of hundreds of different gripping solutions which improves the probability of finding an acceptable gripping solution, but on the other hand, raises the computation time.

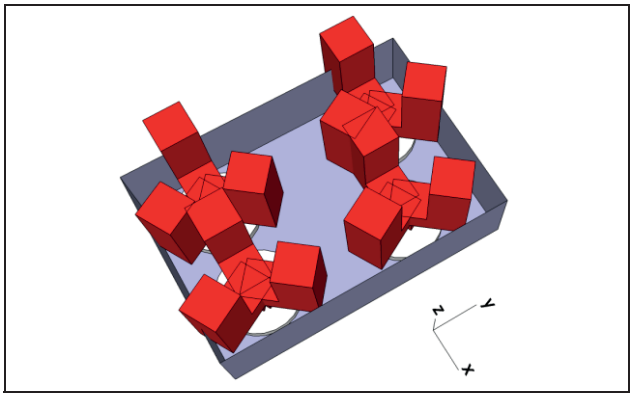

Figure 5 Example of the multiplication of gripping solutions. For each defined gripping point, three different gripping solutions are generated.

4.3 Conditions

Usually a lot of the generated gripping solutions represent impossible grips and would result in e. g. the gripper gripping through the floor of the bin. These impossible grips would be prevented by the collision test described in the next section. To improve the performance of the algorithm and to further eliminate any risk of accepting such an absurd grip, it is also possible to define certain conditions for each gripping point. Only gripping points that pass these conditions will result in a gripping solution being generated. The example of the gripper gripping through the floor, can therefore be prevented by defining a global vector aiming down into the bin and setting a maximum allowed angle between this vector and the z-axis of the TCP. Of course, this is just an example for a condition. A lot of other conditions are thinkable and already used in our industrial deployments.

5 Testing, rating and selection of gripping solutions

5.1 Testing bin collisions

Before the gripping solutions are tested for collisions with the point cloud, gripper solutions that collide with the bin itself are eliminated. This is done by testing collisions between the simplified CAD model of the gripper and the planes that represent the walls and floor of the bin. The exact pose of the bin is already provided by the OPD and requires no extra computation. This test reduces the risk of collisions with the bin in cases with insufficient sensor data on the bin. It also improves the overall performance, since a collision test with the five planes of the bin is a lot faster than a collision test with the point cloud, which will be explained in detail in the next section.

5.2 Point cloud collision test

For the collision test, the point cloud provided by the sensor is transformed into the world frame.
To detect collisions of the gripper with the point cloud, we divide the entire sensor range into three areas which are visualized in **Figure 6**:

- Clear area: This is the space between the points of the point cloud and the sensor origin. No (detectable) obstacles are inside this area.
- Collision area: This is the area where an obstacle is known to be. It only consists of the points in the point cloud.
- Unknown area: This is the area behind the points of the point cloud. It is unknown if there are obstacles in this area.

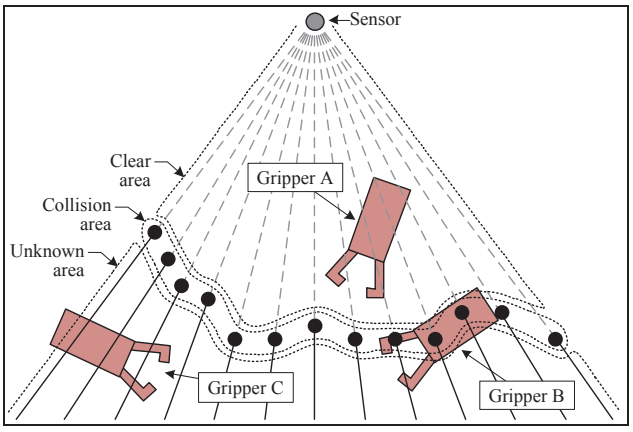

Figure 6 Different areas relevant for the collision test. Gripper A is completely inside the clear area, Gripper B is (partly) inside the point cloud and therefore inside the collision area, whereas Gripper C is completely inside the unknown area.

For safe collision-free gripping, the gripper should be in the clear area at all times. A differentiation between the other two areas is not needed. Therefore, it is also not necessary to test if the actual points of the point cloud are inside the gripper model, which would be more complex. Instead, we take another approach which only tests if a gripper model is (even partly) in the forbidden zone which consists of the collision area and the unknown area. For that, all straight lines from the sensor (origin of the sensor frame) to the measured points are extended beyond these points into the unknown area. This extension (not including the line from the sensor to the point) is now called collision line and should cover the part of the unknown area that the robot can reach.
If a gripper is inside the unknown area (or the collision area) it will now intersect these collision lines as can be seen in **Figure 6**. So testing if a gripper is inside these forbidden areas, can now be realized by testing the intersection of all collision lines with every triangle of the gripper model.

5.2.1 Improvement of performance

To improve the performance of the presented approach not every triangle is tested with every point. Instead, the point cloud is divided in several areas we call boxes, which results in a so called boxed point cloud. This is done by dividing the ranges of all three world coordinates into e. g. 64 parts, which would result in 262 144 boxes.
When testing a triangle for intersections, first, the bounding box of the triangle is determined, representing the minimum and maximum of all x-, y- and z-values. The boxes of the boxed point cloud that may include points inside the bounding box of the triangle can now easily be determined.

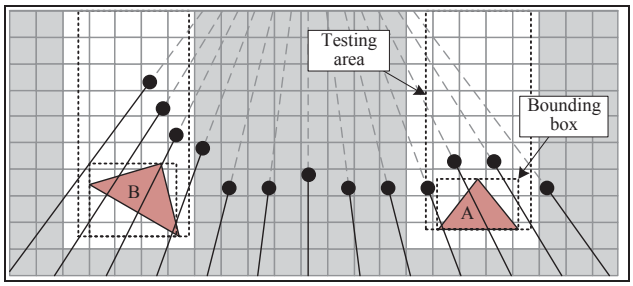

Figure 7 2D example of a boxed point cloud. Displayed are two triangles including their bounding boxes and their testing areas. The points in the white boxes are used for the respective collision tests, whereas the points in the gray boxes are not. For triangle A all relevant points are tested. For triangle B this testing area would be too small because one intersecting collision line is not tested.

If only the points in those boxes are taken into consideration for the collision test, grippers A and B in **Figure 6** can still be tested correctly. Gripper C however is not recognized as a collision since there are no points inside the bounding boxes of its triangles. For this reason, each bounding box is extended upwards to the height of the sensor. That way even gripper C can be recognized as colliding. However, when using a panning sensor, the rays

are not vertical, but slightly tilted. Hence, the bounding box is also extended horizontally to include points that are not exactly above the gripper. The final box, which we call testing area, can be seen in **Figure 7**.

5.2.2 Collision rating

Since there are always tolerances and outliers in real world scenarios, it is not only important to know if any collision line intersects with the gripper, but how many. That way, a rating can be calculated for every gripping solution depending on how many collision lines intersect. The collision rating $r_{collision}$ is calculated with the formula

$$r_{collision} = 1 - \frac{CollisionCount}{divider}$$

where $divider$ is a manually configured parameter representing a reasonable number of collisions which should definitely result in an invalid gripping solution.

5.2.3 Collision test along the path

Obviously the gripper may not only collide with an obstacle at the moment of picking, but also along the path. Although the risk can be reduced by using a good path generation, an additional check is advantageous. For that reason, all (or, to improve performance, only selected) path points are tested in the same way. **Figure 8** shows an example of a collision with the box on the approach path. The final rating for the collision risk is then based on the maximum of the collision count of all tested positions.

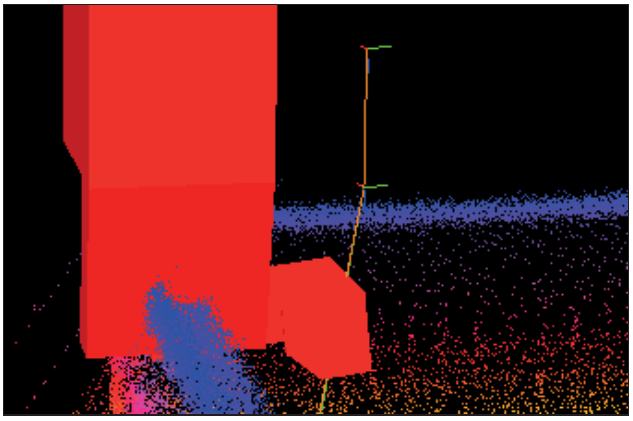

Figure 8 Example of a collision with the box on the approach path. Many points are inside the gripper model at this position. The thin line visualizes the path of the TCP into and out of the box.

5.3 Additional rating influences

Since there may be other considerations than the collision count with the point cloud, a total rating r_{total} is calculated, based on the collision rating $r_{collision}$ and other ratings r_i as follows:

$$r_{total} = r_{collision} \cdot \left(1 - \sum_i w_i\right) + \sum_i (r_i \cdot w_i)$$

The weightings w_i determine the influence of each additional rating parameter. Since the collision rating is obviously an important parameter, the additional weightings are usually close to zero. An example of an additional rating parameter will be shown in the next section.

5.3.1 Object height

To avoid difficult situations where only few gripping solutions exist, there should not be great height differences inside the bin. Therefore it is always preferable to grip higher objects rather than lower ones. This is realized by introducing a rating r_{height} which has a maximum value of 1 for the highest detected object and a rating of 0 for the lowest one:

$$r_{height} = \frac{(h_{object} - h_{min})}{(h_{max} - h_{min})}$$

This height rating influences the total rating as shown above and favours the picking of high objects.

5.4 Selection of the best gripping solution

The gripping solution with the highest rating is now chosen as winner and passed to the robot. However, since there may be no viable gripping solutions at all, the rating has to be above a predefined threshold to avoid collisions in this case. Practical applications showed that, since the algorithm works very robust, the rating difference between valid and invalid gripping solutions is usually very high. Therefore it is quite simple to choose an appropriate threshold.

5.4.1 Multi-level processing

Since a detected object may be partly buried under other objects, it may be difficult to always find a possible position for the gripper. However, this is necessary if a high availability is required. Therefore, in some applications, we employ a multi-level approach, for which each gripping point is given a priority. This way, we can first test the gripping solutions that use the gripping points with the highest priority. If a valid gripping solution is not found among them, we then continue with the next lower priority. Since the gripping points with lower priorities are ignored if a valid solution is already found, this can result in choosing a suboptimal gripping solution. Therefor the highest priority should cover all the usual gripping points. However, if the normal gripping points fail, we have, combined with the multiplication of gripping points, the ability to test hundreds of different gripping solutions.
This approach results in a very low computation time in most cases and still ensures a high availability.

6 Results

The algorithm has successfully been used in several industrial projects including objects with very few gripping points. Once the algorithm was completely configured we encountered almost no more collisions with the box itself or other objects. Only very small or thin objects, like metal bars inside the bin, may not produce enough points in the point cloud to be detected as a collision.
The specifications of the PC that was used for the runtime tests are: Intel(R) Core(TM) i7-2620M Quad Core CPU @ 2.70GHz, 4GB Memory. The installed OS is Windows 7 Enterprise (64 Bit). The boxed point cloud used for the collision test contains around 366 000 points and is divided into 64×64×64 boxes. The simplified gripper model consists of 164 triangles.
The effects of the following features were tested during the runtime tests:

- Collision test with the bin borders as described in section 5.1.
- Abortion of the point cloud collision test once the collision count would result in a rating of zero (in our situation after 350 collisions).
- Testing all path points along the gripping path as described in section 5.2.3. The approach path and the departure path consist of 3 path points each.

Obviously, the runtime depends on the situation inside the bin. **Table 1** shows the results for a common situation where 16 gripping solutions were generated. Four of these gripping solutions intersect the borders of the bin and therefor have a lot of points inside the gripper, whereas ten gripping solutions are completely inside the bin, but may still have slight collisions with other objects. All configurations resulted in the same four valid gripping solutions. Although the testing of bin collisions and especially the testing of the gripping path can help to avoid collisions, this shows that first and foremost these measures improve the performance of the algorithm.

Configuration			**Runtime (in ms)**	
Testing for bin collisions	Abort test at a rating of zero	Test the gripping path	Test of the gripping solutions	Entire algorithm
False	False	False	97.3	105.1
True	False	False	36.9	45.0
False	True	False	32.1	40.3
True	True	False	30.7	38.4
True	True	True	97.0	105.3

Table 1 Runtime results of the algorithm for creating and testing 16 gripping solutions.

As expected, both the testing of collisions with the bin borders and the abortion of the point cloud collision test result in a significantly lower runtime because, without them, a lot of points have to be tested for the gripping solutions that intersect the borders of the bin.

With the best combination of these settings, the algorithm takes an average of just 1.916 ms for testing a single gripping solution (without checking any path points).
Since the gripper is usually not that far inside the bin on the path, testing a path point is typically faster than testing the gripping point itself. Therefore, testing additional 3 approach path points and 3 depart path points for each gripping solution, meaning a total of 112 collision tests, results in a computation time of about 105 ms for the entire algorithm. Therefor the overhead for testing the entire gripping path can be compensated by the mentioned performance measures.

7 Conclusions and outlook

The presented approach provides a working solution to determine gripping points and avoid collisions during bin picking. This is particularly important for objects that have limited gripping areas and for large or complex grippers, especially in industrial environments which require a high availability. Since one of the main focuses of the algorithm is speed, the additional runtime is reasonably low within the entire bin-picking application.
Although the algorithm works very well already, there is of course still scope for improvements. Testing single points along the path obviously has the disadvantage that collisions could still occur between these points. Although the risk could be reduced by automatically creating intermediate points between the given path points, it would be preferable to test the entire path. For the departure path, even the workpiece itself may be included in this test, resulting in a simulated extraction of the object. This could be done by creating a 3D model for the entire path and is a subject of future development.
Furthermore, it would be advantageous to have the ability to adapt the generated path. That way, if a position along the path collides with an object, but the gripping point itself is fine, the path could just be altered or bended slightly to move around any obstacles.

8 Literature

[1] Schraft, Rolf Dieter; Ledermann, Thomas: Intelligent picking of chaotically stored objects. In: Assembly Automation 23 (2003) 1, pp. 38-42.

[2] Tauro, Ricardo A.; Kaiser, Benedikt; Path Planning Process Optimization for a Bin Picking System. In: 41st International Symposium on Robotics (ISR) and 6th German Conference on Robotics (ROBOTIK), 2010, 1 -7.

[3] Buchholz, Dirk; Winkelbach, Simon; Wahl, Friedrich M.: RANSAM for Industrial Bin-Picking. In; 41st International Symposium on Robotics (ISR), and 6th German Conference on Robotics (ROBOTIK), 2010, 1 -6.