Int J Adv Manuf Technol (2006) 30: 540–553
DOI 10.1007/s00170-005-0054-5

ORIGINAL ARTICLE

Cesar Fernandez · Oscar Reinoso · M. Asuncion Vicente · Rafael Aracil

Part grasping for automated disassembly

Received: 2 July 2004 / Accepted: 17 March 2005 / Published online: 17 November 2005

Abstract A robot grasp synthesis algorithm for automated disassembly is presented. The goal is to select grasping points in each part to be disassembled so that a previously planned disassembly sequence can be performed holding the parts firmly and avoiding collisions. The algorithm is structured in five steps in order to make it general enough to cope with different robot grippers and different geometrical data (2D or 3D). The system is learning based, and behaviour rules are automatically extracted from grasping examples given by the user, using mainly decision trees and nearest neighbour techniques. Some simulation experiments have been carried out and results with a two fingered robot gripper are presented.

Keywords Decision trees · Disassembly · Machine learning · Robot grasping

1 Introduction

Governments, enterprises and researchers are recently paying much attention to disassembly processes; the main reason being ecology consciousness [1]. Among the different research topics involved, disassembly automation has been intensively studied [2–5].

The starting point of most automatic disassembly algorithms is the CAD description of the product [6, 7]; sometimes this information is updated or improved with additional data obtained from the actual object to be disassembled, by means of video cameras or other sensing devices [8]. This additional data helps in locating the product pose in the disassembly environment and also in detecting minor deviations from the CAD description.

Using such information, most researches address two problems: first, the generation of a feasible disassembly sequence; and second, the generation of collision-free disassembly paths. However, there is an extra problem that is seldom considered: how to grasp the different parts. Grasp selection has to be considered relevant in a disassembly automation proposal as a wrong grasp may result in the parts being damaged (there may be fragile elements in the part which should not be touched); in the parts slipping from the gripper (non closure grasps, unstable grasps); or in collisions between the grasping device and the product to be disassembled.

The most common grasping device is a robot gripper, and there are different options: the first one is the use of a tool-changer device and ad-hoc grippers for each part to be grasped; and the second option is the use of a versatile multipurpose gripper. This second option seems more adequate as it adds flexibility to the disassembly process, i.e. previously unknown parts can be grasped and previously unknown products can be disassembled. Versatile grippers range from the common two jaw parallel gripper (provided that its stroke is long enough to hold objects of different sizes) to complex multi-fingered robot hands. A survey of grasping devices of different complexity can be found in [9].

Once the gripper is selected, the grasping of an assembly part is defined by the contact points on the surface of the part where to place the gripper fingers (only pinch grasps will be considered in this paper, enveloping grasps [10] are not suitable for disassembly processes as usually there is not enough room as to envelope the part with the robot hand) and the relative pose of part and gripper. There are multiple-infinite-choices, and the selection is by no means trivial.

Most authors that address this problem, simply rely on an iterative algorithm that chooses-randomly or based on simple heuristics-grasping points in order to find a collision free grasp through all the disassembly path using a two jaw gripper [11]. However, other aspects like grasp closure,

C. Fernandez (✉) · O. Reinoso · M. A. Vicente
Industrial Systems Engineering Department,
Miguel Hernandez University,
Elche, Spain
e-mail: c.fernandez@umh.es

R. Aracil
DISAM, Polytechnical University of Madrid,
Madrid, Spain

grasp stability or part damage are not considered. Some other authors [12] adapt techniques originally developed for assembly automation, like the one proposed by Van Holland [10] where finger domains are obtained using both geometrical information from the 3D CAD models and extra information supposed to be available as features (particularly, handling features give information about how to grasp a certain part) or as forbidden areas (areas where the fingers of the robot gripper should not be placed). Another technique adapted from assembly automation is the one proposed by Bard [13] which performs robot hand preshaping from octree based representations of the shape of the part to be grasped. However, all these adaptations do not take into account the higher degree of uncertainty present on disassembly processes, where the products may not match completely their CAD description.

Part grasping has also become a research subject on its own and multiple researchers have proposed different grasp synthesis algorithms. Several approaches can be emphasized: those synthesizing force closure grasps [14, 15], those synthesizing optimal grasps according to different quality criteria [16, 17], or those based on generalized prototypes [18]. However, none of these techniques can be directly applicable to a disassembly scenario, as the part to be grasped is considered to stand alone and restrictions related to the interrelation with other parts of the assembly or with disassembly paths can not be added easily.

2 Problem statement and proposed approach

The goal is to select the optimum grip for a certain part of an assembly, provided that both the disassembly sequence and the disassembly paths have been obtained previously (this work will not focus on such aspects). Both the part to be grasped and the remaining assembly parts are supposed to be defined by a 3D CAD model or by the information extracted from sensor readings. The grasp synthesis algorithm should be general enough as to cope with different grasping devices.

A learning based approach is proposed, where grasp synthesis is performed in five steps. The next sections describe each of these steps:

1. Generation of candidate contact points
2. Evaluation of candidates and filtering of invalid contact points
3. Computation of feasible grasps
4. Evaluation of grasps and selection of the optimum set of contact points
5. Selection of the best arm and hand configuration

3 Generation of candidate contact points

All the points belonging to the outer surface of the part to be grasped not in contact with other parts of the assembly through the disassembly path are considered candidate contact points. These are the points where a finger of the gripper may be placed. This first step can be performed straightforward provided that both an adequate 3D representation of the assembly and full disassembly paths are available. The results (free surfaces) are discretized and converted to a finite set P of z candidate points p_i, as Eq. 1 shows.

$$P = \{p_1, p_2, ..., p_z\} : p_i \in FreeArea \, \forall i \tag{1}$$

4 Evaluation of candidates

4.1 Local attributes selected

Candidate contact points are evaluated on the basis of a comparison of local attributes with previously stored grasp examples. There are two local attributes that have to be measured for each candidate point:

- Distance from the point to the centre of gravity of the object (part to be disassembled).
- Multiresolution measure of the local convexity of the object surface at the contact point.

Using a multiresolution scheme helps in distinguishing valid from invalid contact points as stated in our previous work [19]. In order to create an algorithm as general as possible and adaptable to different disassembly environments, 2D or 3D expressions of the previously defined local attributes are considered.

2D local attributes can be used when the DOF of the robot arm and gripper are limited, so that only normal grasps can be performed, i.e. the gripper is always parallel to the disassembly direction and the contact points are located in a plane normal to such direction. Under this restriction, the part to be grasped can be represented by the outer contour of its projection over the normal plane; and both distances and convexities can be computed in 2D.

When the gripper is equipped with more DOF, it can perform grasps in different directions. Thus, local features have to be computed in 3D, as the orientation of the grasp is not known in advance. Figure 1 clarifies such difference: the 3D geometry of the part to be grasped is not necessary when only normal grasps can be performed.

Local attributes are computed for the z candidate points obtained in the previous step of the algorithm, and they are classified by means of a decision tree inferred from the examples given by the user.

The output of this step of the algorithm is a set P' of z' valid grasping points p_i, where $z' <= z$, as Eq. 2 shows.

$$P' = \{p_1, p_2, ..., p_{z'}\} : p_i \in P \, \forall i, \; p_i \; valid \; \forall i \tag{2}$$

4.2 Machine learning algorithm used

Apart from decision trees, different learning algorithms have been considered (a detailed comparison of their performances can be found in [19]). Both quantitative and

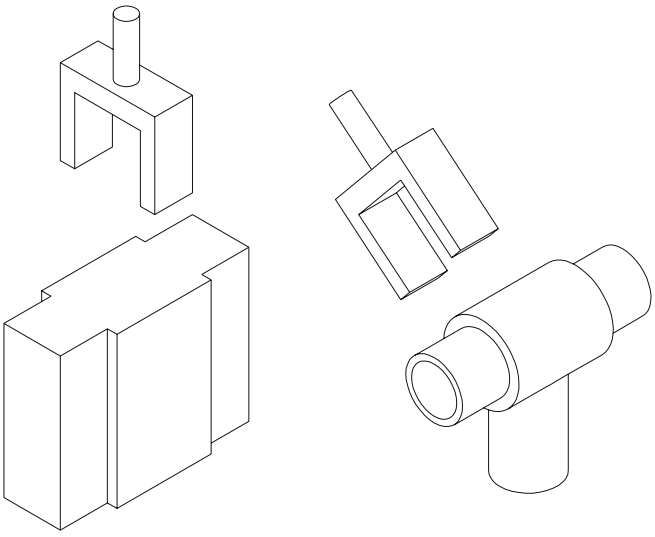

Fig. 1 Left: knowledge about the 3D geometry of the part to be grasped is not needed when only normal grasps are performed: its projection over the plane normal to the grasping direction is enough. Right: when the grasping direction is not known in advance, the 3D geometry is required

qualitative criteria have been used to select the best algorithm for the application. The methods compared were nearest neighbour, multilayer perceptron, radial basis functions, naïve Bayes, decision trees and rule lists.

Four different quantitative criteria were evaluated:

- Percentage of correct classifications in a tenfold cross validation test with 400 examples.
- Percentage of correct classifications with a variable number of training examples ranging from 20 to 380 (allows to evaluate the behaviour of the algorithm when there are few examples).
- Off-line (training) computation time.
- On-line (classification) computation time.

Concerning the first test, the results are shown adjusted to a normal distribution in Fig. 2. The highest classification rates are obtained by nearest neighbour, followed by decision trees, rule lists and multilayer perceptron. The results of the second test are shown in Fig. 3, where the same four algorithms obtain the highest values in all the range of training examples.

The evaluation of computing time brings interesting results: nearest neighbour is not applicable due to its huge on-line computation time, which is linear with the number of training examples. The remaining algorithms have a low and almost constant on-line computation time, which allow all of them to be used. Off-line computation time is not so relevant, as training is only performed once after gathering all the examples. However, only multilayer perceptron and radial basis functions have high values for such time.

The qualitative criteria chosen include readability of the results (relevant as it allows user supervision of the process), robustness to noisy training examples; and sensitivity to tuning parameters (low sensitivity is preferred, as it allows an easier and more reliable implementation in a real application). Table 1 shows the behaviour of the different algorithms under such criteria.

Taking into account all the results, both decision trees and rule lists have to be considered the best algorithms. Among them, the decision trees have been chosen as they have been considered easier to read and to interpret (most relevant attributes are explicitly shown at the higher tree levels).

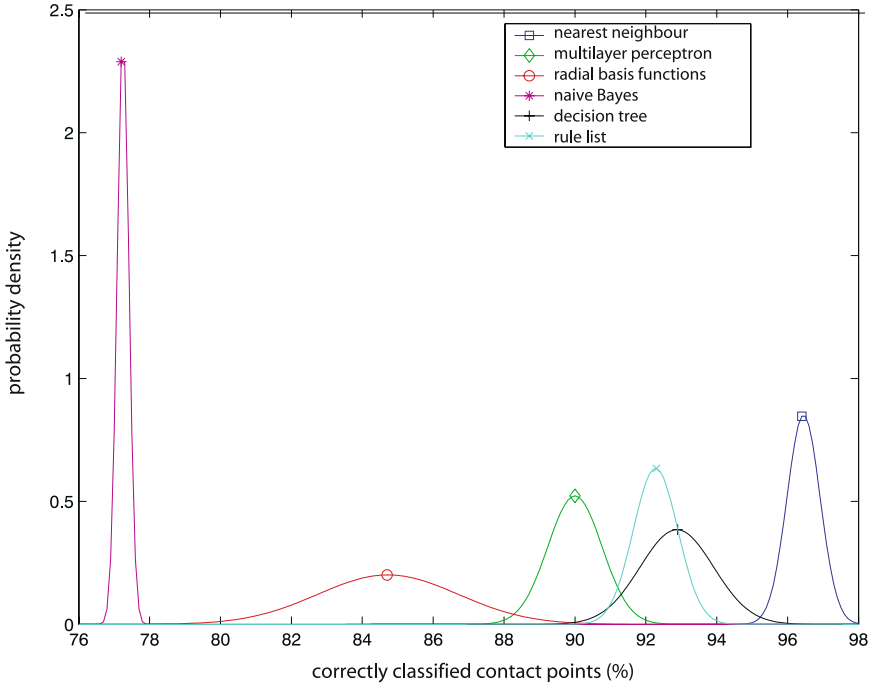

Fig. 2 First quantitative criteria: cross validation results (classification of candidate contact points)

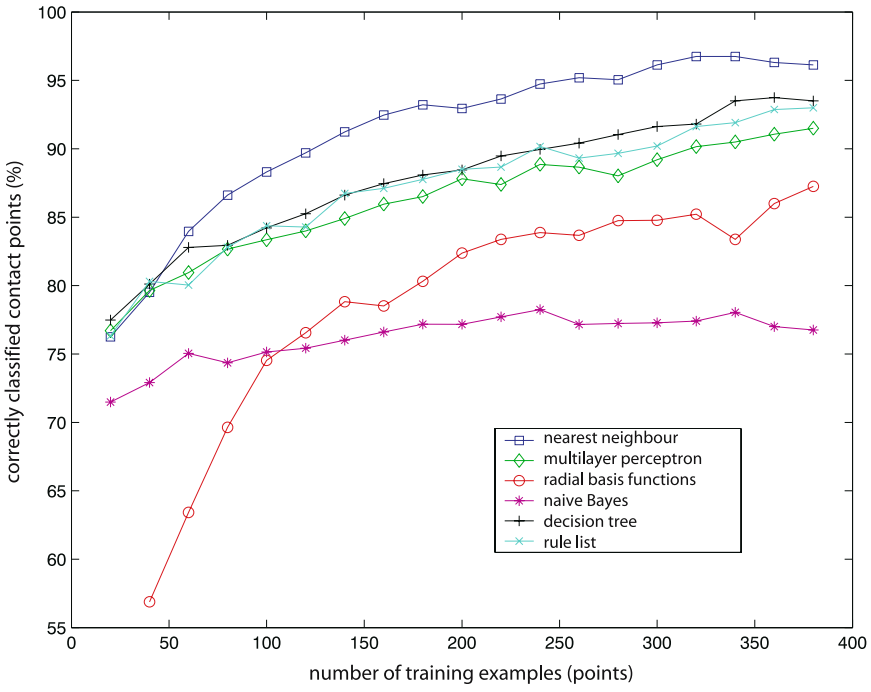

Fig. 3 Second quantitative criteria: results with different number of training examples (classification of candidate contact points)

5 Computation of reachable grasps

Ideally, a generic robot hand or gripper of n fingers may perform $\binom{z'}{n}$ different grasps by placing its fingers over the z' valid contact points. However, most of these combinations of grasping points will either be kinematically unreachable by the robot hand or result in a collision between robot hand and parts of the assembly. The purpose of this step of the algorithm is to filter all these unfeasible grasps.

The first problem to be addressed is the reachability of the contact points. This process is performed quite easily when the grasping device is as simple as a two jaw gripper, but more complex grasping devices make it necessary to solve a redundant and highly dimensional inverse kinematics problem. There are several reasons that justify the complexity of such problem:

- The inverse kinematics must be computed for each of the n fingers of the robot hand. The number of DOF is $m+m_i$, where m DOF correspond to the robot arm and m_i DOF correspond to the ith finger of the gripper. Under such circumstances, the configuration of the robot arm and gripper can be expressed as Q in Eq. 3; where q_i represents the ith joint of the robot arm and q_{ij} represents the jth joint of the ith gripper finger.

$$Q = \{q_1, ..., q_m, q_{11}, ..., q_{1m_1}, ..., q_{n1}, ..., q_{nm_n}\},$$
$$q_i, q_{ij} \in \Re \forall i, j. \tag{3}$$

- As all contact points have to be reached simultaneously, there is an extra restriction that should be added: the arm joint coordinates have to be kept constant through all the n inverse kinematics problems. Figure 4 exemplifies such restriction.
- Apart from the previously mentioned restrictions, there could be additional restrictions imposed by the particular robot gripper used, i.e. some of the joints may be dependant, as it happens in the three fingered articulated gripper shown in Fig. 5.

The previously described problem has not been solved for a general robot gripper with n fingers, each of them with m_i degrees of freedom, connected to the wrist of a generic m DOF robot arm. Some authors propose methods for solving the inverse kinematics in multifingered grippers [20, 21] but these methods can not work in real time, and this problem is really important as the inverse kinematics must be solved $\binom{z'}{n}$ times.

Table 1 Qualitative criteria: the optimum algorithm should have high readability, high robustness to noisy training examples and low sensitivity to tuning parameters

	Readability	Robustness to noisy training examples	Sensitivity to tuning parameters
Nearest neighbour	none	low	low
Multilayer perceptron	low	high	high
Radial basis functions	low	low	high
Naïve Bayes	none	high	low
Decision tree	high	high	low
Rule lists	high	high	low

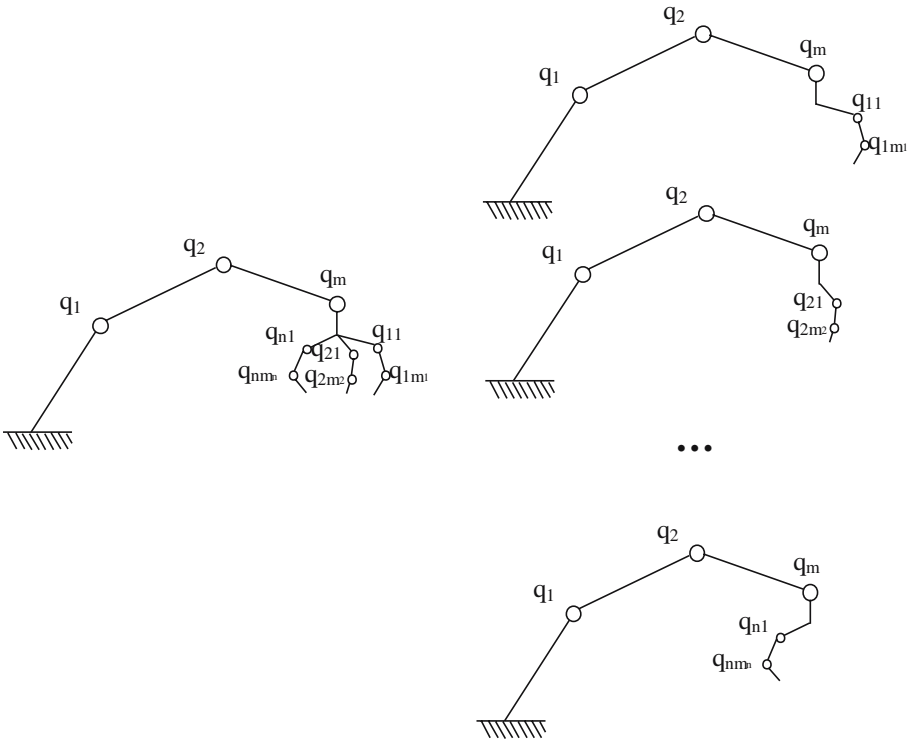

Fig. 4 The m joint values of robot arm are common to all the n kinematic chains (one per gripper finger). The remaining m_i joints belong exclusively to finger i

The second test (collisions check) has been widely studied by different researchers [22]. There are multiple optimized algorithms devoted to the detection of collisions between static or dynamic objects [23–26]. However, even these optimized algorithms are not fast enough to perform the $\binom{z'}{n}$ collision checks required. The main idea is to avoid performing such a huge number of checks.

Considering the two problems to be addressed (kinematics and collisions), a feasibility check strategy is proposed, based on performing successive filterings of increasing complexity. Actually, most of the combinations of contact points can be discarded clearly without the need to perform complex kinematics or collisions analysis, e.g. points too far from each other, points that will clearly result in a collision, etc. Performing simple analysis first, helps in reducing the number of candidates for further stages and thus in reducing computational load. Figure 6 shows the structure of the proposed strategy.

This strategy requires design of specific filters for each particular robot arm and hand. Some experiments are being carried out at present; in the experiments performed so far, the most simple filters deal with the maximum and minimum distances allowable between the contact points, and the distance from the robot origin to the centre of the contact points. These filters are extremely fast and allow discarding many combinations of contact points. Further tests include the computation of the surface normals at the contact points and the establishment of relations between them, and slightly more complex tests deal with simplified collision checks (based on expected angles between surface normal and finger stroke direction). Finally, the full kinematics analysis and detailed collision test are only performed for a small subset of grasp candidates.

The result of this step of the algorithm is the set of reachable grasps where the gripper fingers are placed in valid contact points. Equation 4 represents each possible

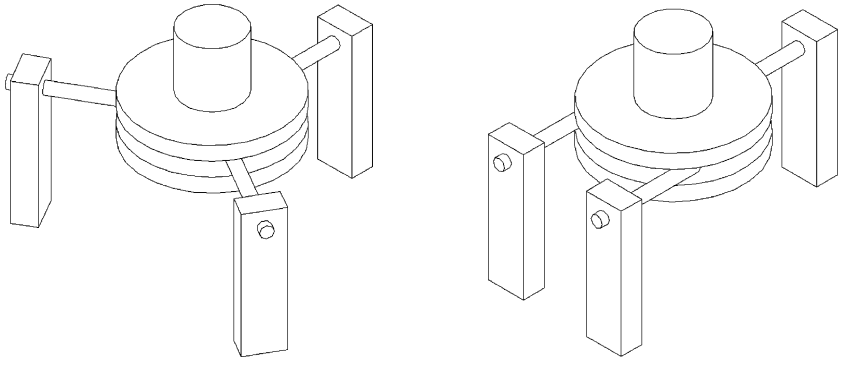

Fig. 5 Three fingered articulated robot hand: an extra restriction is added as the movement of the two articulated fingers is coupled

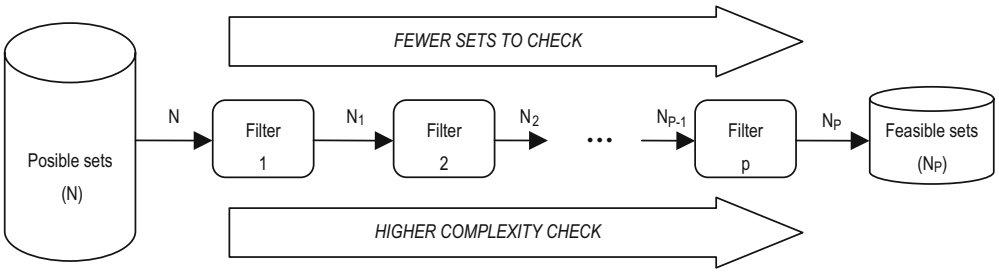

Fig. 6 Successive filterings to reduce computational load

grasp as a set g_i of n valid contact points; the $s = \binom{z'}{n}$ theoretically possible combinations of valid contact points are represented as G in Eq. 5; finally, the s' reachable grasps are represented as G' in Eq. 6.

$$g_i = [p_1, p_2, ...p_n] : p_i \in P', p_i \neq p_j \, \forall i \neq j \tag{4}$$

$$G = \{g_1, g_2, ..., g_s\} \tag{5}$$

$$G' = \{g_1, g_2, ..., g_{s'}\} : g_i \in G, g_i \, reachable \, \forall i \tag{6}$$

6 Evaluation of grasps and selection of the optimum set of contact points

6.1 Global attributes selected

Candidate grasp sets (sets of n grasping points) are evaluated on the basis of a comparison of their global attributes with those of the previously stored grasp examples. The global attributes (not related to a single contact point but to the relations between the n contact points) are measured using as a reference the centre of gravity of the convex hull defined by the n contact points. Two different attributes are defined:

- Distance from the reference point to the centre of gravity of the object.
- Multiresolution measure of the angle between the surface normal at the contact point and the line directed to the reference point.

The first global attribute used allows distinguishing centred grasps from non-centred ones, i.e. it will allow to infer rules showing the kind of objects where the user performs centred or non-centred grasps, and to imitate such behaviour. Figure 7 shows an example of an object being grasped (with a three-fingered gripper) in two different ways, and how the distance attribute allows to distinguish both grasps.

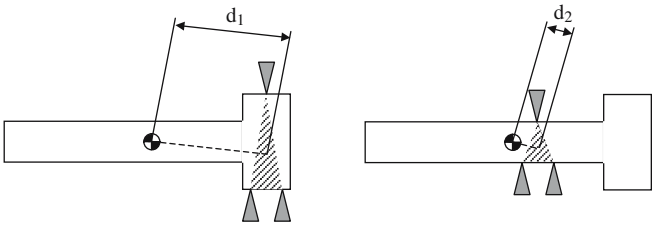

Fig. 7 Distance from the reference point to the centre of gravity allows distinguishing centred from non-centred grasps

The second global attribute has a close relationship with the fulfilment of the force closure condition. It can be proved [27] that a two-fingered grasp (contact points with friction are assumed) fulfils the force closure condition when the line joining both contact points lies within their friction cones. Figure 8 gives examples of grasps fulfilling or not fulfilling such condition.

Under such circumstances, the proposed attribute gives relevant information: as its values approach zero, the grasp is more likely to fulfil the force closure condition.

When a three fingered gripper is used, similar criteria can be established [28]: the grasp fulfils the force closure property if the friction cones of the contact points intersect each other, and the contact normals do not belong to the same half-plane (these are sufficient but not necessary restrictions for determining force closure). Figure 9 shows an example.

As the symmetry axis of the friction cones is collinear with the surface normal, the closer the angle between the normal and the line directed to the reference point gets to zero, the more likely is the force closure condition to be fulfilled.

In brief, such attribute allows to infer rules related to the kind of objects where the user decides to perform force closure grasps, i.e. the situations where a force closure grasp should be chosen.

The proposed attributes are valid for 2D and 3D representations of the objects. Both attributes (distance and

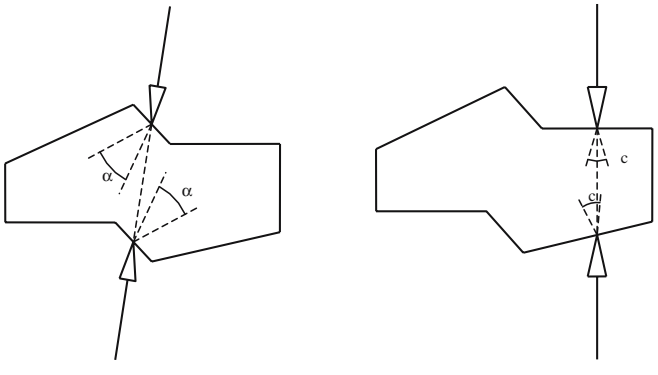

Fig. 8 When two contact points are used, the line joining both points must lie within the friction cones in order to fulfil the force closure condition: in the example, only the right grasp is force closure

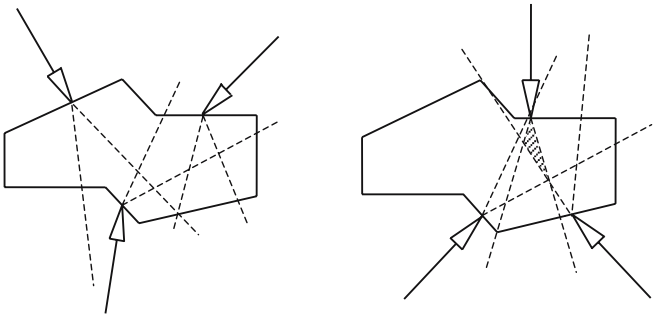

Fig. 9 When three contact points are used, the existence of an intersection between the three friction cones is related to the fulfilment of the force closure condition: in the example, only the right grasp is force closure

multiresolution angle) are measured for all the candidate sets of contact points, and their values are compared to those of the grasp examples provided by the user. Only the candidate grasps with similar attribute values to those of the examples are kept for the next step of the algorithm; the remaining ones are discarded.

The means used to obtain the 3D representation of the objects is out of the scope of the present work, but there could be multiple options:

- First, CAD data of the object could be used. Such data may not be completely accurate due to differences between the design of a certain part and its actual shape, but minor discrepancies do not affect the proposed method, as surface points are discretized to a certain resolution and small details are therefore lost.
- Another option is the use of range sensors, which allow to obtain an accurate 3D representation of the object (however, there could be hidden regions if only a viewpoint is used).
- Finally, video cameras are also a valid option. If a single camera is used, only 2D information can be obtained, so such solution may only be valid for normal grasps.

Otherwise, stereo vision or structured light should be required in order to capture the object 3D shape (again, there could be hidden regions in the object if only a viewpoint is used).

With independence of the means used to obtain the geometry, the proposed approach can be easily applied.

6.2 Machine learning algorithms used

The whole process takes place in two stages. In the first stage, the problem is similar to that of step 2 of the algorithm: the candidate grasp sets are classified as valid or invalid sets. Different learning algorithms can be used for this stage, which is again a classification process. A similar comparison to that performed for step 2 of the algorithm was also carried out, but using global attributes instead of local ones. The classification rates obtained in the cross validation test (first quantitative criteria) are shown in Fig. 10, with slightly different results: the best method is again the nearest neighbour, but the multilayer perceptron outperforms decision trees and rule lists. The effect of the number of training examples (second quantitative criteria) was also analyzed: the results shown in Fig. 11 are quite similar: the best method is the nearest neighbour followed by the multilayer perceptron, but decision trees also offer high classification rates.

The reasons why the results are slightly different can be found in the different distribution of the attribute values.

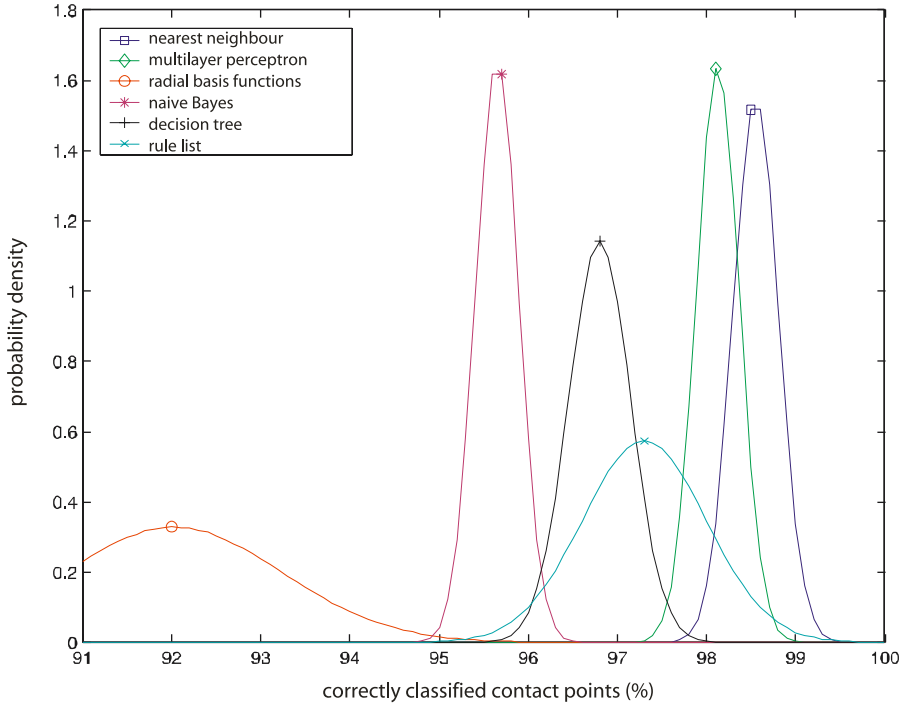

Fig. 10 First quantitative criteria: cross validation results (sets of contact points)

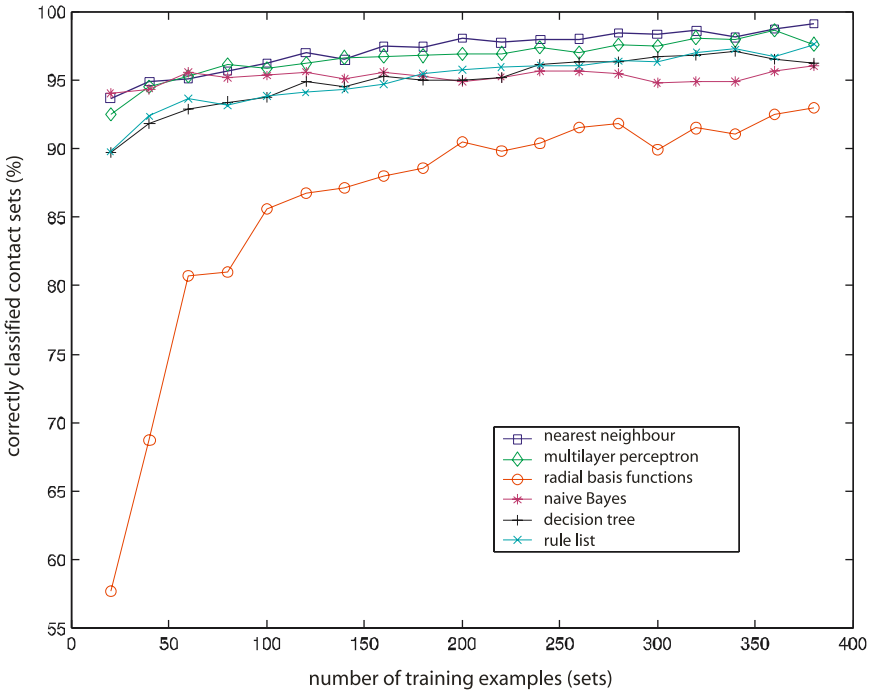

Fig. 11 Second quantitative criteria: results with different number of training examples (sets of contact points)

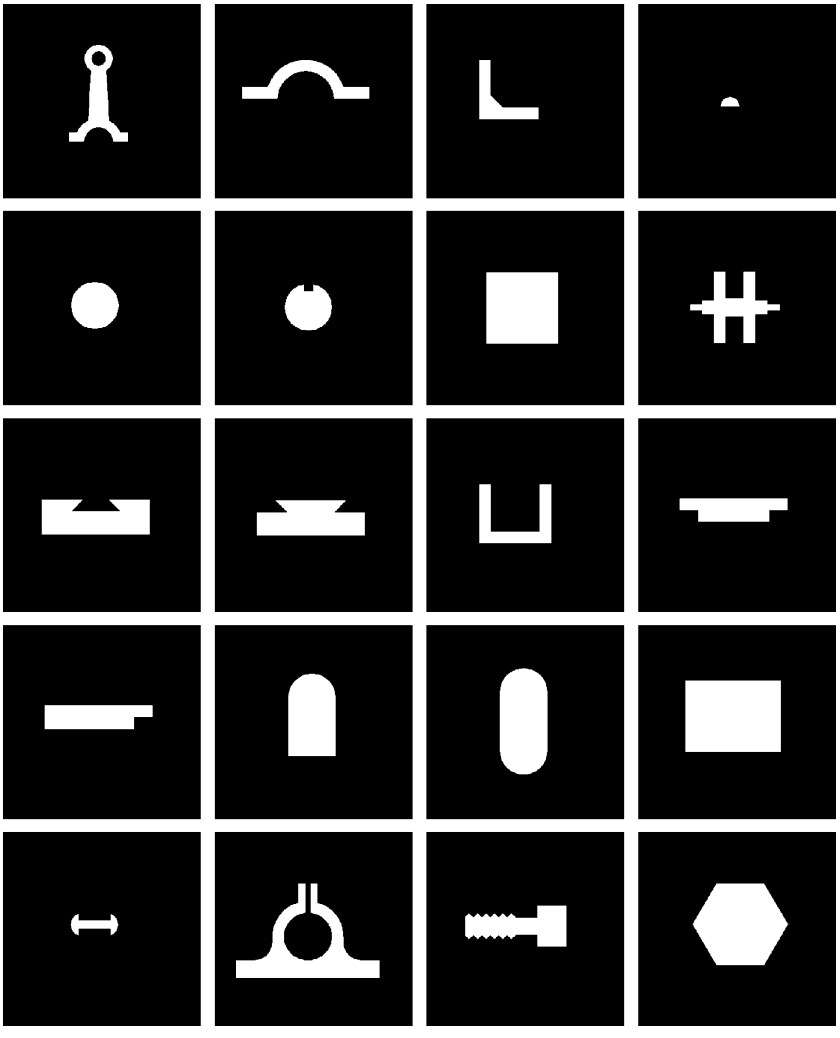

Fig. 12 Some objects of the database created to test the grasp synthesis algorithm

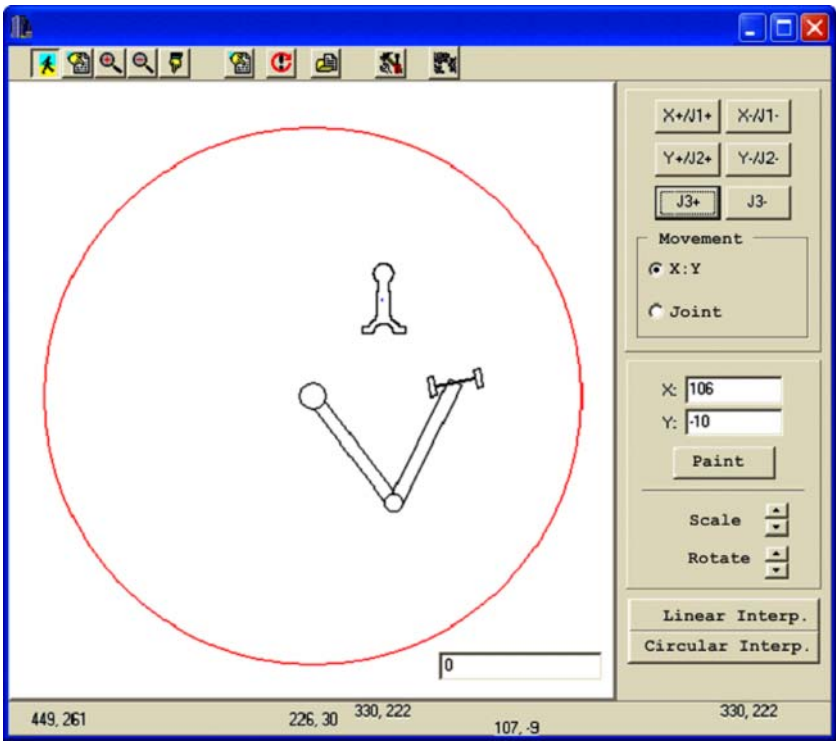

Fig. 13 Simulation environment: general view

Global attributes seem to have easier to classify distributions than local ones, as the percentage of correct classifications in this step of the process is higher than that of step two. Besides, the average sizes of decision trees, rule lists and multilayer perceptrons, which give an idea of the complexity of the decision boundaries, are much lower.

Analyzing the comparative results, decision trees have been selected also for this step of the process, even giving lower classification accuracies than other algorithms: nearest neighbour has to be discarded because of its high on-line computational load; and multilayer perceptron is also considered inadequate as its model is not readable. The best algorithms are thus rule lists and decision trees (their classification accuracies are very similar). Both choices would have been correct, but it was decided to use decision trees because of their higher readability.

The second stage deals with the selection of the best grasp among all the valid ones, as classified by the former decision tree. For this stage, a variation of the nearest neighbour algorithm is used, and the selected grasp is the one closer to one of the valid grasp examples given by the user.

The whole process could have been reduced just to the second stage, choosing directly the feasible grasp more similar to one of the grasp examples. However, this strategy would not be robust to noisy training examples, and a wrong set could be selected due to its proximity to an incorrectly recorded grasp example. That is the reason why a two stage process is proposed instead.

The first stage is described in Eq. 7, where the sets of reachable contact points classified as valid by the decision tree are represented as G″; the second stage is described in

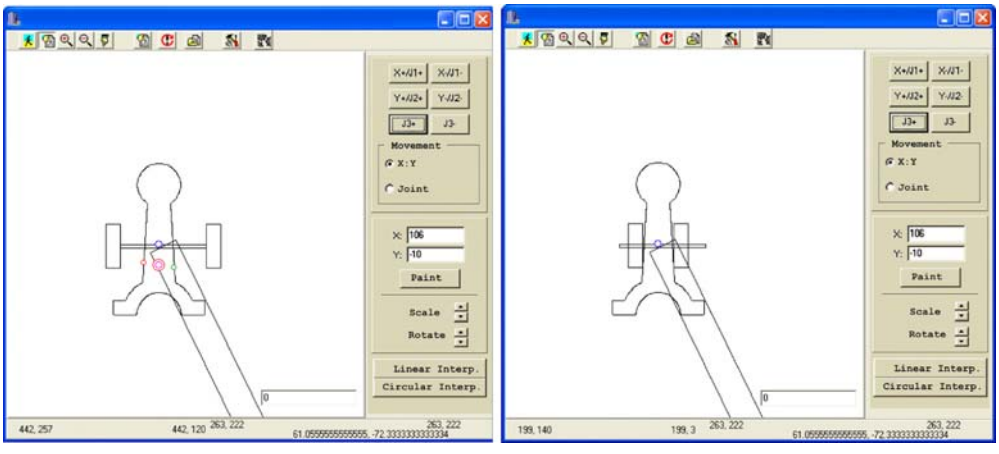

Fig. 14 Simulation environment: grasping a part

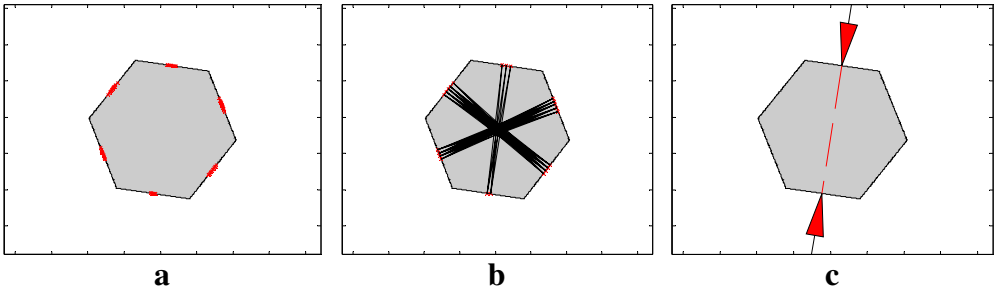

Fig. 15 First part tested: (**a**) valid contact points; (**b**) valid contact sets (2 contact points); (**c**) selected set and synthesized grasp

Eq. 8, where *dist(a,b)* represents the Euclidean distance between grasps *a* and *b*, defined in the space of normalized global attributes; and ej_k represents the *k*th grasp example.

$$G'' = (\{g_1, g_2, ..., g_{s''}\} : g_i \in G', g_i \, valid \; \forall i) \tag{7}$$

$$g_{selecc} = g_i \in G'' : \min_{\forall k} (dist(g_i, ej_k)) \leq \min_{\forall k} \left(dist\left(g_j, ej_k\right)\right) \forall j \neq i \tag{8}$$

The final result of this step of the algorithm is the optimum set of *n* contact points to perform the grasp.

7 Selection of the best arm and hand configuration

Once the contact points have been selected, a configuration for robot arm and hand has to be chosen. In the general situation, there will be multiple (or infinite) choices for reaching the contact points, due to the possible kinematical redundancy of robot arm and hand. In some particular situations (e.g. two jaw parallel grippers attached to a non redundant robot arm) there is only a configuration capable of reaching the contact points, so this step of the algorithm is straightforward.

For the general situation, a learning based strategy is used, in order to find the most similar configuration to that of one of the grasp examples given by the user. A variation of the nearest neighbour algorithm is also used for this step of the process, but an important modification is made: distances are measured in a $m + \sum_{i-1}^{n} m_i$ dimension space, where *m* represents the DOF of the robot arm and m_i the DOF of each of the *n* gripper fingers. However, not all the dimensions of the space have the same relevance in defining a grasp: those joints closer to the contact points are supposed to be more representative of the grasp type. In this way, a weighted version of the nearest neighbour method is used, where higher weights correspond to joints closer to the contact points.

The result of this step of the grasp synthesis algorithm is a configuration of robot arm and hand fulfilling the following properties:

- The optimum set of contact points is reached.
- There are no collisions between robot arm and gripper and the different assembly parts.
- The configuration is optimal in terms of its similarity to one of the stored grasp examples.

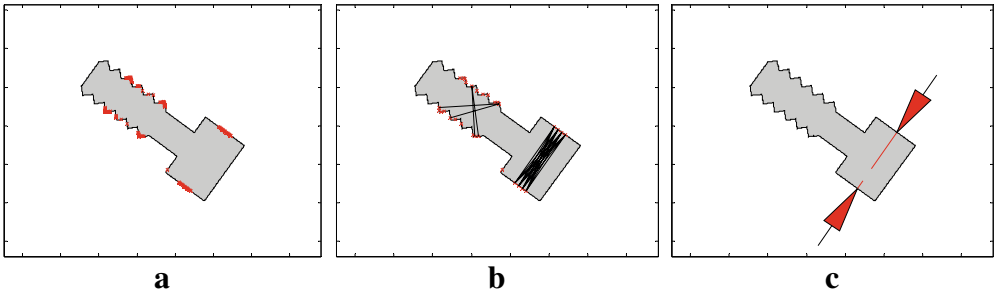

Fig. 16 Second part tested: (**a**) valid contact points; (**b**) valid contact sets (2 contact points); (**c**) selected set and synthesized grasp

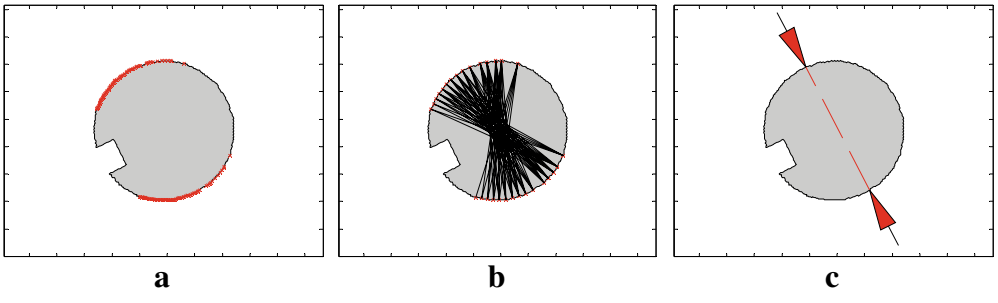

Fig. 17 Third part tested: (**a**) valid contact points; (**b**) valid contact sets (2 contact points); (**c**) selected set and synthesized grasp

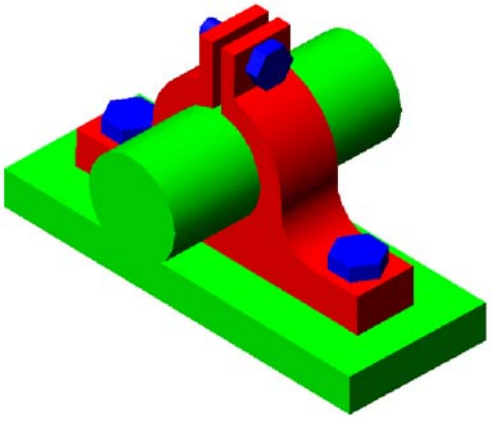

Fig. 18 Assembly example used for the second experiment

8 Experimental results

A database of 50 different parts (partially shown in Fig. 12) has been created in order to test the validity of the algorithm. Some of the parts are available in different sizes, thus increasing the variability of examples used for the tests. Different experiments have been carried out with such a database. First, the validity of the proposed algorithm has been tested for standalone parts: the goal is to test whether the system is able to learn from the examples how to grasp previously unknown parts. Then, the system is tested in an assembly built with different parts of the database, and the goal is to test whether the algorithm is able to select the correct grasp for each part to be disassembled, while avoiding collisions with the remaining parts. A simulation environment with a scara robot and a

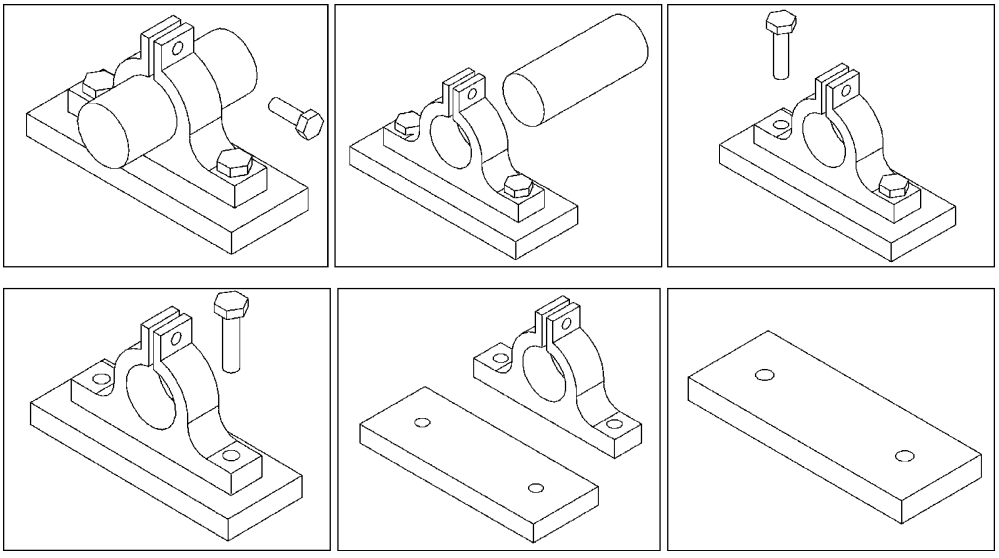

Fig. 19 Disassembly sequence for the second experiment

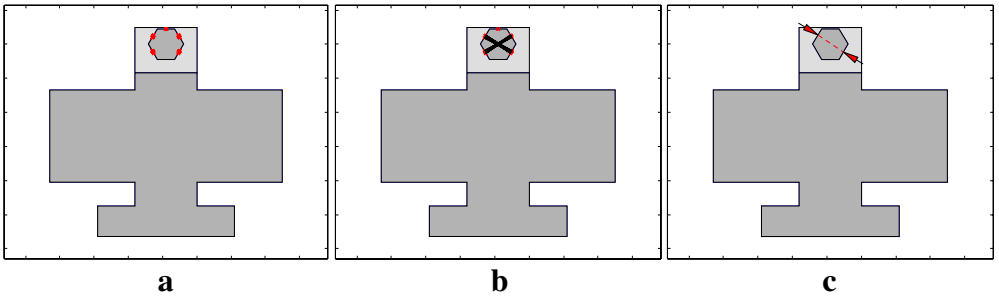

Fig. 20 First part to be disassembled: (**a**) valid contact points; (**b**) valid contact sets; (**c**) selected set and synthesized grasp

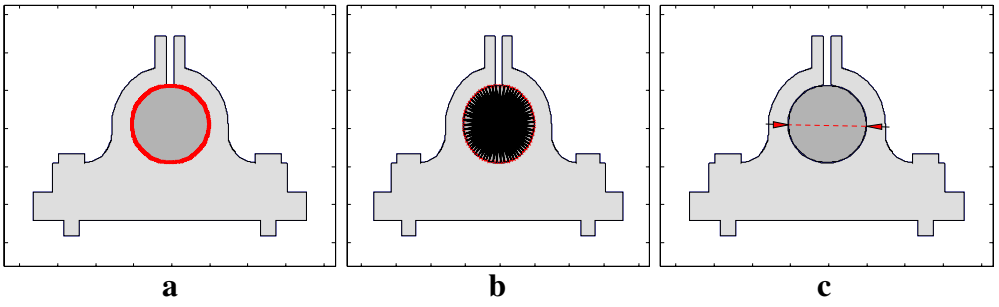

Fig. 21 Second part to be disassembled: (**a**) valid contact points; (**b**) valid contact sets; (**c**) selected set and synthesized grasp

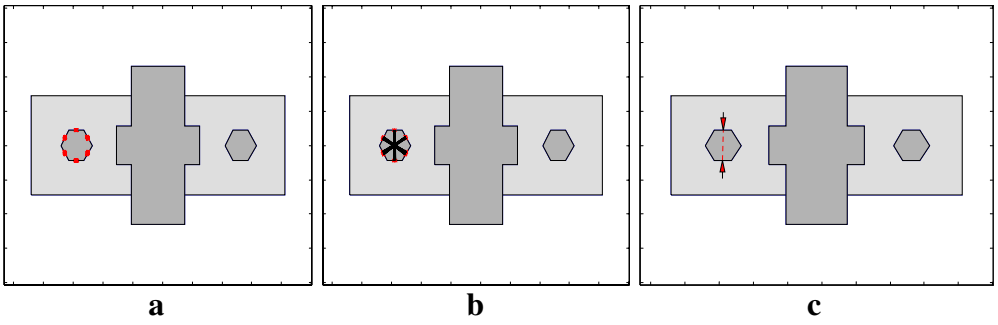

Fig. 22 Third part to be disassembled: (**a**) valid contact points; (**b**) valid contact sets; (**c**) selected set and synthesized grasp (the fourth part is equivalent)

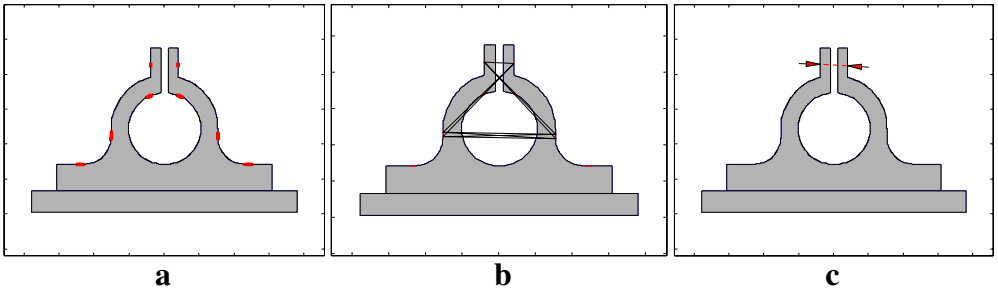

Fig. 23 Fourth part to be disassembled: (**a**) valid contact points; (**b**) valid contact sets; (**c**) selected set and synthesized grasp

two jaw gripper has been developed in order to be able to give grasping examples and test the behaviour of the grasp synthesis algorithm. This environment is shown in Figs. 13 (full view) and 14 (detailed view of the gripper area grasping a part).

8.1 First experiment: standalone parts

The goal of the first experiment is to check the validity of the proposed approach with standalone parts. The simulation environment is used to give training examples to the system. Different parts are presented to the user who should grasp them with the two jaw gripper, and for each example the contact points and the geometry of the part are stored (only 2D information is used, so the geometry is represented as the outer contour of the part).

Obviously, the results improve when the number and variety of the training examples are increased. For this experiment, it was decided to give a total of 60 training examples to the system, randomly selected from the database (the same part may be presented twice). Once these 60 training examples were recorded, the decision trees were inferred from local and global data, and the system was able to perform grasps autonomously. A previously unknown part (not used for training) was then presented to the system, who autonomously decided how to grasp it. Figure 15a shows the valid contact points selected by the algorithm in the contour of a certain part; Fig. 15b shows the valid sets of contact points (2 contact points as a 2 jaw gripper is used); and Fig. 15c shows the final pair of contact points selected and the synthesized grasp. The same results are shown in Figs. 16 and 17 when different parts are presented to the system. It must be concluded that the system is able to grasp autonomously a certain part, and to imitate the user behaviour.

8.2 Second experiment: part grasping in a disassembly sequence

Figure 18 shows the example assembly used for the second experiment. The disassembly sequence and disassembly paths are supposed to be known in advance, and the whole process is shown in Fig. 19. The purpose is to select an adequate grasp for each part to be removed from the assembly. A similar environment to that used in the first experiment will be considered, and only 2D information will be given to the algorithm (outer contour of the parts to be grasped). As only 2D information is used, the z axis of the gripper needs to be fixed a priori; for the experiment it will be considered that the z axis of the gripper is aligned with the disassembly direction of each part, and the goal is to select 2 dimensional grasping points on the contour of the part at a fixed height.

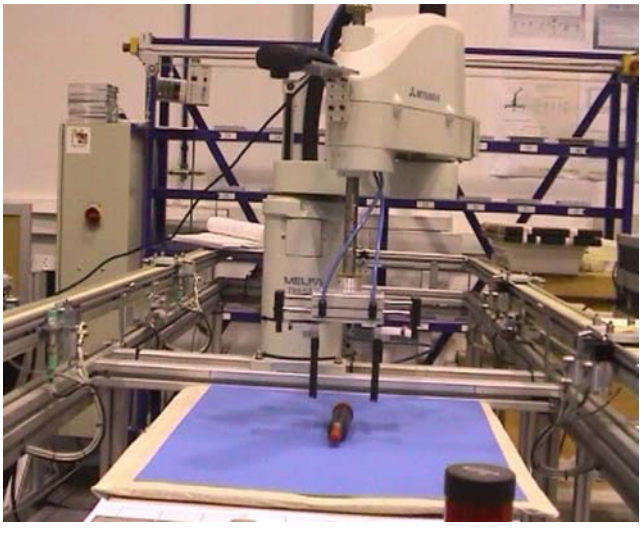

Fig. 24 First tests on a real environment with standalone parts

The algorithm is not retrained for the second experiment, even though the previous training was performed with standalone parts. The intention is to test how the system behaves in previously unknown situations. The results are shown in Figs. 20, 21, 22, 23 for each element to be disassembled. In each figure, the dark grey areas correspond to the 2D projections of the assembly whose height equals or exceeds that of the gripper (i.e. a collision may occur); the light grey regions correspond to the 2D projections of the assembly whose height is below that of the gripper (i.e. there is no risk of collision). The results show that a good quality grasp is found for all assembly parts, and that collisions are avoided.

9 Experimental setup

The same experiments carried out in the simulation environment are being performed at present on a real system. The first tests have shown that standalone parts can be successfully grasped using a single video camera and 2D information. Only the contour of the object is used to select the grasping points, and the objects are grasped at a fixed height. Figure 24 shows an image of the experimental setup.

10 Conclusions

An automated disassembly system must include a grasp synthesis step in order to be applicable in a real scenario.

Learning based grasps synthesis algorithms that imitate the user behaviour are particularly suitable for disassembly processes where the examples given by the user allow to infer relevant information: fragile elements that should not be touched by the grippers, optimum gripper configuration to reach a set of contact points in a certain assembly, etc.

Among the machine learning algorithms compared in this scenario, decision trees are selected: even though their classification accuracy is not the highest one, they offer relevant qualitative advantages like readability and robustness.

Although grasp synthesis is very dependant on the robot arm and gripper used, it is possible to establish a general structure valid in different disassembly scenarios. The five step algorithm proposed can be adapted to many different environments.

Computational load can result in a grasp synthesis algorithm not being applicable in a real scenario. The proposed algorithm decouples local and global grasp attributes, thus resulting in a reduction of the number of grasp candidates to check. Apart from that, a strategy based on successive filterings of increasing complexity is proposed, in order to obtain further reductions in computational load.

Future work will be focused in developing a 3D simulation environment and testing the algorithm in a real scenario. Some preliminary tests have been performed in such a real scenario, with promising results.

References

1. Aksoy HK, Gupta SM (2002) Capacity and buffer trade-offs in a remanufacturing system. Environ Conscious Manuf II, SPIE 4569:167–174
2. Gungor A, Gupta SM (2001) Disassembly sequence plan generation using a branch-and-bound algorithm. Int J Prod Res 39(3):481–509
3. Zeid I, Gupta SM, Bardasz T (1997) A case-based reasoning approach to planning for disassembly. J Intell Manuf 8(2):97–106
4. Zussman E, Zhou MC (2000) Design and implementation of an adaptive process planner for disassembly processes. IEEE Trans Robot Automat 16(2):171–179
5. Torres F, Puente ST, Aracil R (2003) Disassembly planning based on precedence relations among assemblies. Int J Adv Manuf Technol 21(5):317–327
6. Shyamsundar N, Gadh R (1999) Geometric abstractions to support disassembly analysis. IIE Trans 31:935–946
7. Srinivasan H, Gadh R (1998) A geometric algorithm for single selective disassembly using the wave propagation abstraction. Comput Aided Des 30(8):603–613
8. Torres F, Gil P, Puente ST, Pomares J, Aracil R (2004) Automatic PC disassembly for component recovery. Int J Adv Manuf Technol 23(1–2):39–46
9. Bicchi A (2000) Hands for dexterous manipulation and robust grasping: a difficult road towards simplicity. IEEE Trans Robot Automat 16(6):652–662
10. Van Holland W (1997) Assembly features in modelling and planning. Dissertation, Delft University of Technology
11. Puente ST (2002) Desensamblado automático no destructivo para la reutilización de componentes. Aplicación al desessamblado de PC's. Dissertation, University of Alicante
12. Wai Sung RC (2001) Automatic assembly feature recognition and disassembly sequence generation. Dissertation, Heriot-Watt University
13. Bard C, Trocca J, Vercelli G (1991) Shape analysis and hand preshaping for grasping. Proc Intelligent Robots and Systems, Intelligence for Mechanical Systems 1:64–69
14. Nguyen VD (1986) The synthesis of stable force-closure grasps. Technical report AI-TR-905, MIT Artificial Intelligence Laboratory
15. Toth E (1999) Stable object grasping with dextrous hand in three-dimension. Periodica Polytechnica Ser El Eng 43(3):207–214
16. Ferrari C, Canny J (1992) Planning optimal grasps. Proc. IEEE Conf. on Robotics and Automation, Nice, pp 2290–2295
17. Cornellá J, Suárez R (2003) On 2D 4-finger frictionless optimal grasps. 16th IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas, pp 156–162
18. Pollard NS (1996) Synthesizing grasps from generalized prototypes. Proc IEEE International Conference on Robotics and Automation, Minneapolis, pp 901–911
19. Fernandez C, Vicente MA, Reinoso O, Aracil R (2004) A decision tree based approach to grasp synthesis. Proc International Conference on Intelligent Manipulation and Grasping, Genoa, pp 486–491
20. Hwang CS, Takano M, Sasaki K (1999) Kinematics of grasping and manipulation of a B-spline surface object by a multi-fingered robot hand. J Robot Syst 16(8):445–460
21. Hunt KH, Samuel AE, McAree PR (1991) Special configurations of multi-finger multi-freedom grippers: a kinematic study. Int J Robot Res 10(2):123–134
22. Jimenez P, Thomas F, Torras C (2001) 3D collision detection: a survey. Comput Graph 25(2):269–285
23. Lin M (1993). Efficient collision detection for animation and robotics. Dissertation, University of California
24. Gottschalk S, Lin M , Manocha D (1996) A hierarchical structure for rapid interference detection. Proc of ACM Siggraph'96, pp 171–180

25. Hudson T, Lin M, Cohen J, Gottschalk S, Manocha D (1997) V-collide: accelerated collision detection for VRML. Proc. of VRML Conference, pp 119–125
26. Klosowski J (1998) Efficient collision detection forinteractive 3D graphics and virtual environments. Dissertation, University of New York
27. Nguyen VD (1986) The synthesis of stable force-closure grasps. Technical report AI-TR-905, MIT Artificial Intelligence Laboratory
28. Ponce J, Faverjon B (1995) On computing three finger force-closure grasp of polygonal objects. IEEE Trans Robot Automat 11(6):868–881