2013 IEEE/ASME International Conference on
Advanced Intelligent Mechatronics (AIM)
Wollongong, Australia, July 9-12, 2013

Contact State Based Representation of Object Relations in the Environment for Dexterous Manipulations

Susanne Petsch and Darius Burschka

Abstract— **Robots which are supposed to replace a human worker need sophisticated manipulation capabilities. Such capabilities are required for demanding everyday tasks as well as for specialized operations. This includes, e.g., dexterous manipulations in robot-assisted minimally invasive surgery.**

Sophisticated manipulation tasks require not only appropriate physical capabilities from the hardware, but also advanced knowledge about the dexterous manipulation task itself. Hence, we propose an abstract representation of dexterous manipulation knowledge. It is based on a contact state perspective of the environment. Objects in the environment are described relative to their specific type of contact with the environment. This enables a robot-independent reuse and adaption. The task can even be processed if the environment changes (e.g., due to occurrence of obstacles) of if another robot has to be used.

Our experiments illustrate the flexibility which the representation provides. Obstacle avoidance and efficiency of the manipulator's motion are taken into account in the scenarios.

I. Motivation

Sophisticated manipulation capabilities are an important step towards daily use suitability for robots. This poses not only requirements on the physical capabilities of the involved robotic hand and arm. Advanced knowledge about dexterous manipulation is necessary as well. We focus on the representation of such knowledge in this paper. The representation has to enable a successful reuse of the knowledge even if the environment changes. Hence, it should be possible to build up alternative procedures to achieve the desired goal. Moreover, the knowledge representation should be independent of the robot to allow a knowledge usage on different robotic systems. The representation should be easy to use, since a human should be able to command the robot without programming the exact path. The required knowledge for the representation should be extractable from observation. To conclude, a representation at an abstract level is required rather than, e.g., a record of observations from demonstrations.

The aim of a task is usually a desired goal state of objects in the environment. The performance of the task requires, then, a modification of the current state of the environment. In general, a state could be related to a physical state of an object. For example, a cup can be empty or it can be filled with coffee. Hence, its physical state (weight) is different. Moreover, the handling properties of an object can differ (e.g., tilting of a cup filled with coffee should be avoided). A representation of such properties has already been presented in [1]. We want to go a step further in this paper. We focus on more dexterous tasks, which require a much more detailed knowledge about the goal state in the environment. The representation of such tasks should be flexible enough to describe pre-defined fixed paths, if desired. At the same time, less limited paths should be represented appropriately as well. It should be possible to change those paths, if there are other actors or objects in the environment, which affect the scope of action of the robot.

Fig. 1. The desired contact state is water in the cup. Hence, one could directly fill water into the cup (top figure). Of course, one could use a tool, e.g., a can inbetween to achieve the desired contact state (bottom row). The final choice depends on several factors like efficiency (usage of an additional tool requires time capacities) or further tasks (e.g., refilling of the cup).

Possible applications are daily life manipulations as well as specialized tasks. An example of such a specialized task are medical applications (robot-assisted minimally invasive surgery). Knot-tying is an example of a demanding task in this area. This requires very complex manipulations of the involved threads. Such manipulations do usually not take place in the same pre-determined environment. The environment can change each time. Hence, the manipulation path has to be adapted. Moreover, it should be possible to transfer knowledge from one medical robot to another. This could be done by a direct transfer of the knowledge base or by an observation of a robot demonstrating the tasks.

We use the perspective of *contact states* as basic of the

This work was supported by a cooperation with the German Aerospace Center (DLR).

Susanne Petsch and Darius Burschka are with the Machine Vision and Perception Group, Department of Informatics, Technische Universität München. 85748 Garching, Germany
{petsch,burschka}@in.tum.de

abstract representation here. We describe the environment and its changes as contact states and contact states changes. Such a contact state contains not only the information about the existence of a contact, but also about the type of contact. Therefore, we introduce appropriate relations between objects. These can describe the type of contact in more detail, if complex contact like knots are desired. The contact state perspective enables a clearly arranged representation with a wide range of descriptions from simple to complex contacts. The abstraction of knowledge in the described contact states allows flexibility in the task execution through alternative paths. The representation is independent of the robot. A transfer from one robot to another is easily possible.

An illustrative example is shown in Fig. 1. If somebody wants to drink water, he could ask for a cup with water. The aim contact state is, hence, water in the cup. Of course, one could directly fill water from the water conduit into the cup. The usage of a tool like a can inbetween is another possibility: The can could be filled with water and the water can be filled from the can into the cup. Both ways (with and without the can) lead to the desired final contact state. The advantages or disadvantages of the tool usage can depend on further *external factors* like efficiency (reduced efficiency due to an additional tool) or further tasks (re-filling of the cup at another place).

We define a *tool* as appliance to change the contact states in the environment to the desired ones. The robot itself is defined as an *active object* which executes the required manipulations. All other objects which cannot perform actions themselves are *non-active objects*. Tools are, in general, also non-active objects, which are used actively to change the contact states. For example, if somebody wants to drink water from a cup, we can consider a can as tool to fill water in the cup.

It is possible to depict very simple manipulations in the contact state perspective. For example, an object can be transported from one location to another. The aim contact state is, then, the contact of the object with its final location. Hence, the *Functionality Map* of [1] can be easily used. Moreover, the contact state perspective has advantages concerning the execution of transportation tasks. For example, box A has to stand on box B on a desired table T. Currently, both boxes are placed separately on another table. Our perspective of contact states allows to build up of two alternatives: Box B can be transported to T and placed there. Afterward, A can be transported and put on B. Alternatively, A can be placed on B, first, and, then, both boxes can be transported and placed on the table T. The final choice of the procedure depends on external factors (e.g., heavier load, but just one transportation in the second procedure).

To sum up, we present a representation of the environment for dexterous manipulations. The representation is chosen in a manner, such that

1) it extends existing work with respect to a detailed information representation for dexterous manipulation,
2) it is clearly structured to enable an easy usage,
3) it is reusable even in changing environments, and
4) path alternatives can be build up if feasible.

The paper is structured as follows: After an overview of the related work, the approach is presented. In the approach, the contact state and the role of the involved objects are defined. Moreover, the composition and the usage of the knowledge are described. Afterward, we depict the advantages of our approach in exemplary scenarios. We end with conclusions and future work.

II. Related Work

The changing contact state (sometimes also contact ”mode“) between an object and a robotic hand has been analyzed several times. For example, in [2], a randomized manipulation planner was presented for a multi-fingered hand and switching contact modes. The contact state between a robot and an object (passive, hybrid and active closure) was discussed in [3]. A contact space dynamic model and active joint space model were developed in [4]. A manipulation of an object using the entire body was discussed in [5]. Whole body contact manipulation methods were presented in [6], [7]. For example, the method in [7] allowed the physically demanding task of, e.g., transferring a patient from a bed to a wheelchair.

Moreover, approaches regarding a contact between an object and its environment have been proposed. For example, the kinematics analysis of such a manipulation was presented in [8]. A contact state transition graph for graspless manipulation was proposed in [9]. Srinivasa *et al.* [10] considered the contact between the robot and the object as well as the contact between the object and the environment in control synthesis for dynamic contact manipulation.

In contrast to existing work, we do not focus on the contact between a robot and its environment, respectively, objects. Our perspective has the aim to describe the environment by the contacts within the environment. In contrast to assembling tasks, we present a general representation for dexterous manipulation in the environment. Just the current and the desired contact state descriptions are known here. This distinguishes our work also from [11]. There, sequential constraints of tasks were learned through a Programming by Demonstration approach. The process of the manipulation itself was in the foreground there. Our starting point is the aim state of the manipulation. Moreover, we focus on a representation for very dexterous manipulations.

Our work in this paper is inspired from the abstract representation of actions in [1]. In contrast to [1], we focus here on more dexterous manipulation, which requires a much more detailed knowledge about the manipulation and the desired aims.

Possible applications of our approach include dexterous manipulations in medical applications. A good overview of surgical and interventional robotics can be found in [12], [13], [14]. An abstract representation of surgical tasks was, e.g., suggested in [15] based on [1]. It is important to distinguish our work from the aim of surgical skill teaching in general. Reiley and Hager [16] used Hidden Markov Models (HMMs) for surgical skill evaluation of robotic

minimally invasive surgery. Discrete HMMs were built at task level and at level of surgical gestures. The analysis of different skill levels requires an efficiency check of procedure planning. Padoy *et al.* [17] proposed Workflow-HMMs for monitor the workflow based on 3D motion features. They used a hierarchical HMM with phase-probability variables, in order to model the dependencies between distinct phases an the top level and to model the dependencies within individual phases. The statistical modeling and recognition of surgical workflow was presented in [18]. In [19], a Human Machine Collaborative (HMC) system was proposed to perform portions of surgical tasks autonomously and other parts manually. HMMs were used to learn subtasks, which are performed autonomously later on.

Our representation is independent from the used robotic system. The authors in [20] validated robotic surgery training assessment across different training platforms. We do not focus on a *certain execution* of a task with a *certain robot*, but on the abstract description of a dexterous manipulation task itself. Our aim is an abstract representation of *dexterous* manipulations in *general*. Medical procedures are one area of possible applications. It is important to point out, that this description is not considered to be directly used for automated surgical operations. It is an abstract representation of required capabilities for a sophisticated manipulation.

III. Approach

In the following, the contact state and the roles of objects are defined for the representation of the environment. The composition and usage of the contact state knowledge are described afterward.

A. Contact State Definition

We define a contact state with respect to the involved objects and properties of the contact:

```
contact(objects, contact_properties)
```

The contact properties can be described in different ways. This enables a flexible usage under different requirements. For example, a set of contact points can strictly describe the contact state. Another, much richer and more complex possibility are relations. Relations have the advantage, that they can describe the contact properties in a more abstract and less restricted manner. We define a relation in contact properties by a relation attribute and the involved objects:

```
relation: [relation_attribute, objects]
```

Such relations allow alternative procedures, since the path is described by the *properties* of the dexterous manipulation. Motion-restrictions or efficiency requirements can be considered. For example, a search for alternative paths can be processed (e.g., due to additional obstacles in the environment).

The usage of relations or fixed points depends on the application. There can be tasks, for which a path of fixed points is necessary as for a precise cut along a line. Other tasks allow several or even many paths to achieve the desired goal state, even if the manipulation is dexterous. For example, if a knot has to be tied at shoes, the procedure is not limited to one fixed path. Instead, the complex contact state of the involved threads can be described by relations (as we will also see later on).

In general, it is possible to define the contact properties as set of properties, which can, in turn, contain a set of properties. This allows to structure the properties. As the term ”set“ already indicates, the properties of the contact can be reordered arbitrarily within the set. In contrast, a relation is strictly defined, since the adjective describes the relation of the objects to each other. Hence, the order of the involved objects matters in the relation.

B. Object Role Definition

The objects involved in the task can be active like a robot with a gripper (*active objects*). Other objects are not active themselves, as, e.g., a cup (*non-active objects*). Of course, it is possible to use non-active objects through an active object. We call these non-active objects *tools*. The role of the objects in the contact states can depend on the current task. For example, a can can be used as a tool to fill a cup with water. If the task is the contact state ”water in the cup“, we are done. Otherwise, the cup can, in turn, be a tool, e.g., to drink water.

The contact state perspective allows the composition of alternatives in a task. One or more tools can be included in the task flow, if desired. We have already illustrated such an example (water drinking, see Motivation and Fig. 1). The final choice of a procedure depends on external factors, which can, e.g., be stored in the *Atlas* presented in [21]. Our definition of contact states and the subsequent definition of the object role enables the build up of alternatives if desired and appropriate.

C. Composition and Usage of the Contact State Knowledge

The described knowledge representation through contact states can be composed through different procedures. One procedure is the contact state definition by humans. It is not necessary to program the entire robot. Just the above descriptions need to be filled out. A second procedure is the observation of demonstrations by humans or other robots. The contact states in the environment have to be observed, then. Hence, we are independent of the demonstrator (physical build-up and capabilities of the demonstrator, knowledge representation, etc.).

In order to achieve possible paths which can fulfill the task in a scenario we need to evaluate the contact state description. If the task allows just a path of fixed points, the path is pre-determined. It can be executed straight forward. If the properties are described by a relation, the relation has to be evaluated first, before a path can be created. As we described, a contact property can contain sets of properties. An element of these properties can, in turn, consist of a set of properties. Therefore, we need to consider several levels of properties. We start with one of the properties at

the lowest level. This means, that these properties are not defined through another property set.

If the contact properties are defined by a set, the execution order of the elements in the set is not defined in advance. This could, e.g., be determined by a real experiment or a physical model, which simulates the execution of the relations in different orders. The results of such a simulation, respectively, experiment show, which execution orders lead to the desired contact state. It is possible, that one, several or all possible orders lead to the goal.

The objects themselves can already determine the area of possible paths. One object or parts of an object (e.g., a thread) can be fixed to a certain position. The task has to be processed within the reachable region of the (partly) fixed object. The movable object(parts) can be moved to achieve the desired goal. Moreover, the movable object(parts) allow to provide alternative paths. If, e.g., a knot has to be tied at a shoe, the possible intersection point of the threads is limited to the intersection area of both threads. During the knot-tying execution, the intersection point is flexible within this intersection area. At the end, the threads are tightened and the position of their intersection is much more limited to a smaller region.

The contact property and relation based description is not limited to pre-defined paths. The objects can be described relative to other, possibly movable objects. Hence, the final path can be chosen under consideration of temporary constraints (e.g., other organs as obstacles). Different approaches can be used to build up the final path. We simply take a path which fulfills the desired properties and constraints, here. In future work, the choice of the final points can, e.g., be optimized as suggested in [22]. Another possibility is the usage of Dynamic Movement Primitives (DMPs) [23], [24].

IV. Experiments

In order to illustrate the advantageous options of the contact-based perspective, we build up three scenarios. Some of these scenarios require detailed knowledge about the corresponding dexterous manipulation. Two scenarios are in the context of medical applications. One further scenario can also be applied in the medical context: knot-tying. We describe the properties of the knot as for a "normal" knot, which is usually used by humans to tie their shoes. Such a scenario illustrates a possible non-medical, daily-life application.

Our scenarios focus on a top-down approach for the desired tasks. This means, that the task knowledge is described in the contact states and the corresponding relations (e.g., filled in by a human user). As described, a bottom-up approach could be applied as well.

We simulate the experiments. Real-world applications would require many advanced sensing methods to illustrate our approach (e.g., check of the depth of a cut in scenario I of thread detection at the beginning of scenario II). This is not the focus of this paper. The implementation is done in C/C++. The inverse kinematics is estimated through a stochastic approach for global minimization [25]. We use the implementation by Oliver Ruepp [26]. This way of estimating the inverse kinematics allows us the estimation of several alternative joint configuration sequences, since we are neither limited to pre-defined start configurations of the robot nor to possible local minima along the path. Further details can be found in [27].

We describe the robotic system in the DH-convention suggested by Denavit and Hartenberg [28] in the form shown in [29]. We simulate an arm of the MiroSurge system from DLR [30] with a gripper [31], [32], since its DH-parameters are available in [30], [31], [32]. The robot's end-effector has to pass an entry point to the human body. This entry point has to be considered during the entire procedure. In general, any medical robot could be used in the scenarios, since the approach is independent of the robot.

A. Scenario I: Cut

The first scenario is a simple cut on an organ during an operation. Such a cut is performed by a knife. Hence, the contact **cut_on_organ** is defined as follows:

```
cut_on_organ({knife, organ},
    cut_properties)
```

The **cut_properties** define, how the cut on the organ is performed. A cut cannot be chosen arbitrarily in an operation. It has to be performed accurately along a certain path, which is represented by a set of successive points. Of course, a cut has a certain depth, which defines how deep the cut has to be.

```
cut_properties = {cut_points, depth}
```

The path consists of fixed points. Hence, it is not possible to find alternatives. However, we can optimize the configurations of the robot along this path, e.g., with respect to efficiency in control. We evaluate different robot configurations sets for this scenario in our simulations. The configuration set which has the smallest maximal speed peak along the path has a peak at about 0.15 rad per time unit and an average speed of 0.05 rad per time unit. The configuration set with the minimal average speed (0.04 rad per time unit) has a slightly higher maximal speed peak at 0.16 rad per time unit. The differences regarding the speed peak and the average speed are relatively small in this experiment.

B. Scenario II: Knot-tying

The second scenario is a knot-tying scenario. Two threads are fixed at certain points and a knot has to connect both threads. Consequently, the contact **knot_tying** involves a set of threads and certain properties, which define the type of contact (the knot).

```
knot_tying(thread_set, knot_properties)
```

Theoretically, more than two threads could be knotted. We focus here on two threads, since it shows the basic principle.

```
thread_set = {t_1(fp_1, l_1),
              t_2(fp_2, l_2)}
```

Each thread t_i has its respective fix point fp_i and a length l_i. The properties of the knot are defined by two sets of properties. The knot has a desired intersection point ip and the knot should be tightened at this intersection point.

```
knot_properties = { intersection_point ip,
    tightened_knot at ip }
```

The intersection point is the point of contact. There, the knot is further characterized by the properties of the type of knot. A knot of a simple, daily-life shoe tying scenario is used here:

```
intersection_point ip = {
  [over_behind, (t_{1, lower}, t_2, fp_1)],
  [under, (t_{1, upper}, t_2)],
  [under, (t_{1, upper}, t_{1, lower})] }
```

with

```
[over_behind, a, b, c)]
```

as description of putting a over and behind b. The variable c can be seen as point of perspective to define ”behind“ an object.

A knot is tightened through a distance maximization for each thread tip and the intersection point, respectively, the opposite fixed points.

```
tightened_knot = {
  [maximize distance, (t_{1, tip}, ip)],
  [maximize distance, (t_{2, tip}, ip)],
  [maximize distance, (t_{1, tip}, fp_2)],
  [maximize distance, (t_{2, tip}, fp_1)],
  tolerance t_k}
```

The tolerance t_k increases the area around the points, where the described properties are fulfilled.

In order to achieve paths which can fulfill the task in this scenario, we need to evaluate the contact state description. First, the intersection point is of interest, since it is at the lowest level in the property description. A simulation of the possible manipulation orders would, e.g., show, that, if the last relation (move the upper part of t_1 under its lower part) is perform first, there will be no long-term effect, since further actions of t_1 will destroy the effect. Therefore, it has to be processed later. If the upper part of t_1 is put under t_2 first, it will be impossible to move the lower part of t_1 over t_2 stably. Hence, the relations have to be achieved in increasing order. Consequently, we cannot get alternatives regarding the execution order of the relations.

We see in the definition of the intersection point, that two threads are involved. At least one of them has to be moved to change the contact states. We assume here for simplicity, that we use two manipulators. Each manipulator moves just one of the threads. Thread t_1 should be moved over and under t_2. This means, that a lift of t_1 and t_2 is necessary. t_2 can be lifted before or during the procedure (last possibility: before t_1 should be moved under t_2). Hence, these two alternatives are possible.

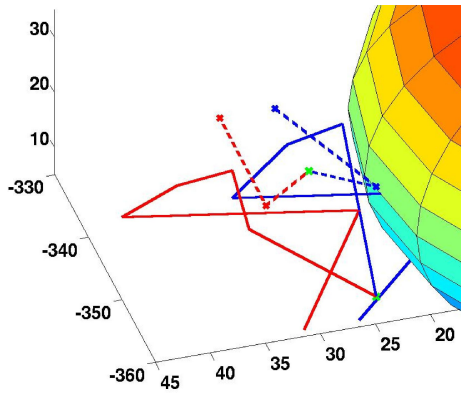

Fig. 2. Original and shifted path in the knot-tying scenario. The original path (blue) cannot be applied, when the colored organ is present, since both are intersecting. In contrast, the shifted path (red) is collision-free. The dashed lines illustrate the path of the supporting manipulator (lifting one thread, tightening). The green crosses indicate the pick-up points of the threads.

The execution of the property tightened_knot is straight forward, but not fixed to a pre-defined path. The corresponding distances have to maximized within the tolerance t_k.

In our simulation, one path is created without further constraints. A second path is built under the consideration of an obstacle - another organ. This requires a partial shift of the first (original) path (see also Fig. 2). The start positions are the same for both paths, whereas the intermediate points are shifted away from the organ. The end points are just slightly moved to achieve an end position, which is similar to the original one, while avoiding the obstacle. The advantage of our representation is clearly visible here: Only the pick-up points of the threads are fixed. The final execution of the desired manipulation is not limited to a fixed path. A possible path can, e.g., be shifted easily to avoid an obstacle.

The differences between the original and the shifted path are small regarding the average speed and the maximal speed peak (average: about 0.7 and 0.6 rad per time unit; peak: both 3.1 rad per time unit). The average speed of the supporting manipulator (lifting the thread, tightening) is the same in the original and the shifted path (0.1 rad per time unit). The peaks in the speed profile differ (0.3, respectively, 0.5 rad per time unit). To conclude, the shift of the path influences the efficiency of the manipulation just partly. The obstacle avoidance can be performed without disadvantages regarding the efficiency of the main manipulator, whereas the speed peaks in the profile of the supporting manipulator differ.

C. Scenario III: Suturation

The third scenario describes a suturation during an operation. A suturation is performed between at least two objects, which have to be in contact at certain points afterwards. The contact **suturation** consists of the tissues, which have to get in contact under consideration of the suturation properties.

```
suturation(suturation_tissues,
           suturation_properties)
```

We consider two objects o_1, o_2 for our suturation scenario, which have a respective stiffness st_i. Moreover, an additional tool ("supporting_tissue") is introduced to keep the objects o_1, o_2 in contact.

```
suturation_tissues = {o_1(st_1),o_2(st_2),
                      supporting_tissue}
```

Theoretically, the objects can be set in contact in any way. For example, the end-effectors of a two-armed robot could move the objects together. If the end-effector unhand its contact to the respective object, the objects might jump back into their original positions without contact. Hence, another tool like a thread is needed to keep the objects in a long-term contact. Theoretically, one could bind the thread around both objects. If the thread cannot be put around the objects, the thread can connect both objects through a stitch. We want to achieve the contact of the **suturation** through stitches along a desired contact line. A sufficient number of suturation_points has to ensure this contact.

```
suturation_properties = {
  suturation_points,
  stitch_properties }
```

The stitches at the suturation_points are, in turn, described through certain **stitch_properties**. For example, a simple direct stitch from one tissue to the other can be chosen. A stitch in form of a cross is more demanding, but more stable. Moreover, a stitch into one object has a certain direction, which can be more or less restricted. We choose a simple stitch for our experiments here:

```
stitch_properties = {
  {simple_stitch},
  {penetration_direction}}
```

The contact **suturation** illustrates our concept of the role of the involved objects. In order to perform the desired stitch, a needle is necessary. The needle and the thread are non-active objects, which are used as tools. The thread is the additional tool ("supporting_tissue") to keep the objects in contact. The needle is a further tool, which is necessary during the execution. The tissues which have to get in contact are non-active. The two-armed robot is the active object, which performs the task.

In our simulation, we assume that the tissues are soft and deformable. No handover of the needle is required. We treat the needle as elongation of the manipulator as long as it is grasped by the end-effector. The path of the end-effector is restricted at the respective stitch points. However, it is less restricted in the approach phase and in the after-stitch phase. The closer the path point to the stitch point is in these phases, the more restricted it is. Hence, we use a cone-shaped path restriction in the approach and after-stitch phase. The cone's height points towards the penetration direction. We represent the cone by points on the cone's cover (see Fig. 3) and a step-wise adaption of the orientation to the desired orientation. If restrictions have to be considered, alternatives can be chosen within the cone. For example,

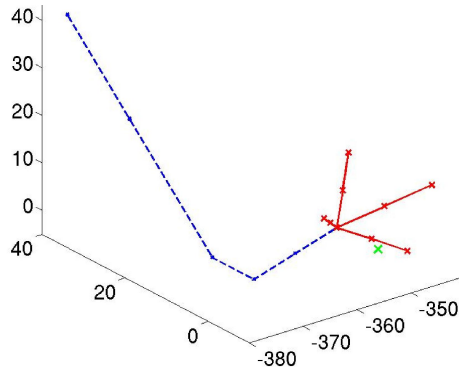

Fig. 3. Path of the suturation scenario. The blue path depicts the fixed part of the path in the scenario. The red part shows alternative paths in the approach phase, which is less restricted. The green cross indicates the center of an obstacle (e.g., another organ). The entire obstacle is illustrated in Fig. 4.

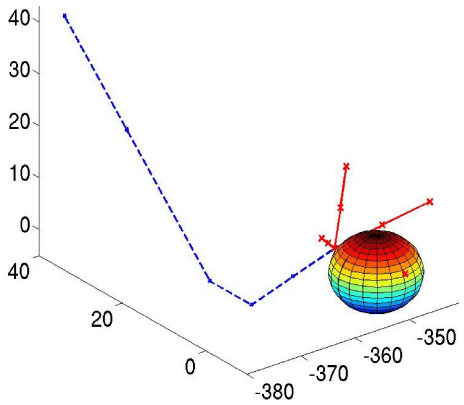

Fig. 4. Path of the suturation scenario with depicted obstacle (Similarly to Fig. 3).

other organs should not get injured by the needle tip. In order to decrease injury avoidance, one could choose the path with the maximal distance to other organs. The organs could be represented through virtual fixtures [33], [34]. The alternative paths in cone-shape are similar to principles in [35]. Parts of our path are fixed (stitch points), whereas others are more flexible (approach phase).

First, the configuration set with minimal average speed (0.21 rad per time unit; peak: 1.26 rad per time unit) is determined for the fixed part of the path (the stitch points). Its first configuration along the fixed path is the aim configuration at the end of the approach phase. Four possible approach paths to the first fixed point are analyzed. These four paths form the described cone (see Fig. 3). The system analyses correctly, that the bottom path cannot be used, since it intersects the organ. The top path is furthest away from the organ. Fig. 4 illustrates these results clearly. Both paths at the side have the same distance to the organ. The path on the left has the smallest average speed (1.1 rad per time unit) and the smallest speed peak of all four paths (2.6 rad per time unit). The average speed of the top path is just slightly worse with 1.2 rad per time unit. Its speed peak at 2.9 rad per time unit is higher.

V. Conclusion and Future Work

The proposed representation of object relations in the environment for dexterous manipulations is based on a contact state perspective. This allows not only an easy handling of complex dexterous manipulations, but also reusage and adaption in changing environments if feasible. Our scenarios illustrate the advantages. The paths are adapted if necessary due to obstacles (other organs). Moreover, efficiency criteria have been included in the final choice of the path. The usage of different robots is possible, since the representation is robot-independent.

In future work, a specific path optimization approach would be desirable, e.g., similarly to [22]. Moreover, the discrete set of possible paths could be extended to a continuous set in the future.

References

[1] S. Petsch and D. Burschka, "Representation of manipulation-relevant object properties and actions for surprise-driven exploration," in *IEEE/RSJ IROS*, San Francisco, California, USA, 2011, pp. 1221–1227.

[2] M. Yashima, "On planning for whole arm manipulation with switching contact modes," in *IEEE/RSJ IROS*, 2001, pp. 1596–1601.

[3] T. Watanabe, K. Harada, T. Yoshikawa, and Z. Jiang, "Towards whole arm manipulation by contact state transition," in *IEEE/RSJ IROS*, 2006, pp. 5682–5687.

[4] G. Liu, J. Li, and Z. Li, "Coordinated manipulation of objects by multi-fingered robotic hand in contact space and active joint space," in *IEEE ICRA*, 2002, pp. 3743–3748.

[5] K. Harada and M. Kaneko, "Whole body manipulation," in *IEEE International Conference on Robotics, Intelligent Systems and Signal Processing*, 2003, pp. 190–195.

[6] K. Salibury, W. Townsend, B. Eberman, and D. DiPietro, "Preliminary design of a whole-arm manipulation system (wams)," in *IEEE ICRA*, 1988, pp. 254–260.

[7] T. Mukai, S. Hirano, M. Yoshida, H. Nakashima, S. Guo, and Y. Hayakawa, "Whole-body contact manipulation using tactile information for the nursing-care assistant robot riba," in *IEEE/RSJ IROS*, San Francisco, California, USA, 2011, pp. 2445–2451.

[8] Y. Aiyama and K. Sato, "Kinematics analysis of manipulation of environment-contacting object with free-joint-structure," in *IEEE International Conference on Mechatronics and Automation*, 2011, pp. 40–45.

[9] Y. Aiyama, T. Yasui, and T. Arai, "Planning of object motion by graspless manipulation using contact state transition graph," in *4th IEEE International Symposium on Assembly and Task Planning*, 2001, pp. 184–189.

[10] S. Srinivasa, M. Erdmann, and M. Mason, "Control synthesis for dynamic contact manipulation," in *IEEE ICRA*, 2005, pp. 2523–2528.

[11] M. Pardowitz, R. Zöllner, and R. Dillmann, "Learning sequential constraints of tasks from user demonstrations," in *5th IEEE-RAS International Conference on Humanoid Robots*, 2005, pp. 424–429.

[12] P. Kazanzides, G. Fichtinger, G. Hager, A. Okamura, L. Whitcomb, and R. Taylor, "Surgical and interventional robotics: Core concepts, technology, and design," *IEEE Robotics and Automation Magazine*, vol. 15, no. 2, pp. 122–130, 2008.

[13] G. Fichtinger, P. Kazanzides, A. Okamura, G. Hager, L. Whitcomb, and R. Taylor, "Surgical and interventional robotics: Surgical cad-cam systems," *IEEE Robotics and Automation Magazine*, vol. 15, no. 3, pp. 94–102, 2008.

[14] G. Hager, A. Okamura, P. Kazanzides, L. Whitcomb, G. Fichtinger, and R. Taylor, "Surgical and interventional robotics: Surgical assistance systems," *IEEE Robotics and Automation Magazine*, vol. 15, no. 4, pp. 84–93, 2008.

[15] D. Burschka and O. Ruepp, "Vision-based analysis of conventional surgical procedures," in *IEEE/RSJ IROS: Workshop on Methods for Safer Surgical Robotics Procedures*, San Francisco, California, USA, 2011.

[16] C. Reiley and G. Hager, "Task versus subtask surgical skill evaluation of robotic minimally invasive surgery," in *Medical Image Computing and Computer-Assisted Intervention -MICCAI 2009*, 2009, pp. 435–442.

[17] N. Padoy, D. Mateus, D. Weinland, M.-O. Berger, and N. Navab, "Workflow monitoring based on 3d motion features," in *IEEE International Conference on Computer Vision Workshops, IEEE Workshop on Video-oriented Object and Event Classification*, 2009.

[18] N. Padoy, T. Blum, S.-A. Ahmadi, H. Feussner, M.-O. Berger, and N. Navab, "Statistical modeling and recognition of surgical workflow," *Medical Image Analysis*, vol. 16, no. 3, pp. 632–641, apr 2010.

[19] N. Padoy and G. Hager, "Human-machine collaborative surgery using learned models," in *IEEE/RSJ IROS*, San Francisco, California, USA, 2011, pp. 5285–5292.

[20] Y. Gao, M. Sedef, A. Jog, P. Peng, M. Choti, G. Hager, J. Berkley, and R. Kumar, "Towards validation of robotic surgery training assessment across training platforms," in *IEEE/RSJ IROS*, San Francisco, California, USA, 2011, pp. 2539–2544.

[21] S. Petsch and D. Burschka, "Estimation of spatio-temporal object properties for manipulation tasks from observation of humans," in *Proceedings of the IEEE ICRA*, Anchorage, Alaska, USA, 2010, pp. 192–198.

[22] S. Petsch and D. Burschka, "Path optimization for abstractly represented tasks with respect to efficient control," in *Proceedings of the IEEE ICRA*, Karlsruhe, Germany. To appear, May 2013.

[23] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Movement Imitation with Nonlinear Dynamical Systems in Humanoid Robots," in *IEEE ICRA*, Washington, DC, USA, 2002, pp. 1398–1403.

[24] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and Generalization of Motor Skills by Learning from Demonstration," in *IEEE ICRA*, Kobe, Japan, 2009, pp. 763–768.

[25] C. Papazov and D. Burschka, "Stochastic global optimization for robust point set registration," *Computer Vision and Image Understanding*, vol. 115, December 2011.

[26] O. Ruepp, "Recovery of structure and motion from monocular images under poor lighting and texture conditions," Ph.D. dissertation, Technische Universität München, 2012.

[27] S. Petsch and D. Burschka, "Estimation of inverse kinematics of arbitrary serial chain manipulators and human-like robotic hands," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Wollongong, Australia, July 2013. Accepted.

[28] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *Journal of Applied Mechanics*, pp. 215–221, 1955.

[29] J. J. Craig, *Introduction to Robotics - Meachnics and Control*. Prentice Hall, 2005.

[30] U. Hagn, T. Ortmaier, R. Konietschke, B. Kübler, U. Seibold, A. Tobergte, M. Nickl, S. Jorg, and G. Hirzinger, "Telemanipulator for remote minimally invasive surgery," *IEEE Robotics and Automation Magazine*, vol. 15, no. 4, pp. 28–38, 2008.

[31] U. Seibold, B. Kübler, and G. Hirzinger, "Prototype of instrument for minimally invasive surgery with 6-axis force sensing capability," in *IEEE ICRA*, 2005, pp. 496–501.

[32] S. Thielmann, U. Seibold, R. Haslinger, G. Passig, T. Bahls, S. Jörg, M. Nickl, A. Nothhelfer, U. Hagn, and G. Hirzinger, "Mica - a new generation of versatile instruments in robotic surgery," in *IEEE/RSJ IROS*, 2010, pp. 871–878.

[33] L. Rosenberg, "Virtual fixtures: Perceptual tools for telerobotic manipulation," in *IEEE Virtual Reality Annual International Symposium*, 1993, pp. 76–82.

[34] A. Kapoor, M. Li, and R. Taylor, "Constrained control for surgical assistant robots," in *IEEE ICRA*, 2006, pp. 231–236.

[35] O. Brock and O. Khatib, "Executing motion plans for robots with many degrees of freedom in dynamic environments," in *IEEE ICRA*, vol. 1, 1998, pp. 1–6.