Project Acronym:	GRASP
Project Type:	IP
Project Title:	Emergence of Cognitive Grasping through Introspection, Emulation and Surprise
Contract Number:	215821
Starting Date:	01-03-2008
Ending Date:	28-02-2012

Deliverable Number:	D13
Deliverable Title :	Temporal grounding of primitives and adaptive grasping primitives
Type:	PU
Authors	V. Kyrki, J. Laaksonen, A. Morales and J. Felip
Contributing Partners	LUT, UJI

Contractual Date of Delivery to the EC:	28-02-2010
Actual Date of Delivery to the EC:	28-02-2010

Contents

1 Executive summary **5**

A Attached papers **7**

Chapter 1

Executive summary

Deliverable D13 presents second year developments within workpackage WP3 "Self-experience of Grasping and Multimodal Grounding". According to the Technical Annex of the project, D13 presents activities connected to Tasks 3.1, 3.2, and 3.3. The objectives of these tasks are defined as

- **[Task 3.1] - Control Architecture.** Initially, a hierarchical control architecture will be defined and developed such that it allows relating the concepts of the grasping ontology defined in WP2 to the immediate control. After the architecture has been defined, this task will continue with the definition and development of the general control architecture components, mainly a Cartesian controller and high-level supervisory and visual controllers.
- **[Task 3.2] - Multimodal Grounding.** The task aims for the definition and development of a grounding mechanism connecting action primitives and attributes with uncertain sensor information, including modelling of the uncertainties involved. Initially, the modelling of uncertainties of the three sensor types (visual, tactile, proprioceptive) is studied considering the context of the attributes of the grasping ontology. Later, the task will continue by studying the temporal grounding problem as a state estimation problem with uncertain information, as the concepts and therefore the symbol set are defined by the grasping ontology.
- **[Task 3.3] - Robust action primitives.** The task aims for the definition and evaluation of adaptive and robust control approaches for individual action primitives. The main focus will be on studying the possible grasp primitives for different hand kinematics (parallel jaw, three-fingered, five fingered) and to identify robust parameterisable primitives through evaluation. Parameterisation of the primitives allows self-experience to be used for improving the performance during future attempts.

The work in this deliverable relates to the following second year milestone:

- **[Milestone 5]** - Implementation of high-level controllers including a global uncertainty model, integration and evaluation in the simulator and experimental platforms, grounding grasping primitives.

The progress in WP3 is presented briefly below, and in more detail in the appendix containing attached scientific publications.

- Attachment A presents the control architecture developed in Task 3.1. The architecture addresses embodiment independent sensor-based control of manipulation. Manipulation skills are represented using an abstract representation which allows transferring knowledge over different embodiments. The hardware independence is demonstrated by using two different GRASP demonstration platforms with different sensory capabilities to perform the same task using same instructions. Current on-going work includes the implementation of the architecture on other GRASP platforms. Integration of the control architecture with the simulator is described in D17.
- Attachment B presents a sensor-based adaptive grasping primitive developed in Task 3.3. The primitive is used to perform power grasps on previously located unknown objects. Only few assumptions are made about the objects, related to the their size and bounding box. The primitive

takes as input the starting position, orientation and preshape of the robot hand. Using this input, a sensor-based controller adapts to the unknown shape of the object and corrects the uncertainty in the object pose. The primitive is tested successfully over a wide variety of objects, including those from the GRASP object set, and with different types of errors in input. A preliminary version of the report was attached to the first year deliverable (D5).

- Attachment C presents that machine learning can be used to link tactile sensor measurements to the abstract concept of "stable grasp", in other words, to connect sensor measurements to the abstract concepts used by the control architecture, as defined in Task 3.2. The detection of grasp stability is an example of a difficult case of temporal grounding of primitives, as the grasp stability is not in general a directly observable variable. Using machine learning for connecting sensory image with abstract concepts allows the use of the embodiment independent control architecture developed in Task 3.1. The paper also proposes to use simulation for the generation of training data for the learning process. A future goal is to use the sensor simulation developed in WP6 and integrate that with the approach.

- Attachment D explores the idea of stability detection from haptics further (Task 3.2). The report concentrates on two issues: First, which features are useful for the task, and second, what kind of a machine learning method is preferable for the instantaneous recognition, that is, performing the recognition using measurements from a single time instant. Also, more extensive experiments on several platforms of the project are reported.

In addition to the results presented above, there is on-going work on the following topics:

- Development of more adaptive grasping primitives to cover the entire grasping and manipulation process (Task 3.3). The primitives are then used along with the simulator developed in WP6, with the goal to orchestrate the sequence of primitives by comparing measured and predicted contact events. The work is inspired by the neurobiological framework proposed by Flanagan et al. (Current Opinion in Neurobiology, 2006).

- Estimation of inertial properties of grasped objects by force sensor data for anchoring physical attributes (Task 3.2). This allows transferring these physical properties of an object under manipulation to the world model in the simulation engine.

Appendix A

Attached papers

A Janne Laaksonen, Javier Felip, Antonio Morales and Ville Kyrki. Embodiment independent manipulation through action abstraction. To be published in *IEEE International Conference on Robotics and Automation, ICRA 2010*.

B Javier Felip and Antonio Morales. Robust sensor-based grasp primitive for a three-finger robot hand. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*.

C Yasemin Bekiroglu, Janne Laaksonen, Jimmy Jorgensen, Ville Kyrki and Danica Kragic. Learning grasp stability based on haptic data. Submitted to *Robotics: Science and Systems, RSS 2010*.

D Janne Laaksonen, Ville Kyrki and Danica Kragic. Evaluation of Feature Representation and Machine Learning Methods in Grasp Stability Learning. Submitted to *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2010*.

Embodiment Independent Manipulation Through Action Abstraction

Janne Laaksonen, Javier Felip, Antonio Morales and Ville Kyrki

***Abstract*— The adoption of robots for service tasks in natural environments calls for the use of sensors to allow manipulation of objects under imperfect environment knowledge and the use of knowledge transfer from humans. This paper addresses these challenges by proposing a new abstraction architecture for embodiment independent sensor-based control of manipulation. The aim is to address three specific challenges: hardware independent control of manipulation, use of sensors to alleviate problems of complexity and uncertainty of the environment, and ease of transferring knowledge over different embodiments through a hierarchical abstract representation of manipulation skills. The proposed abstraction architecture is demonstrated for hardware independence and failure detection on two different manipulator platforms.**

I. INTRODUCTION

One of the important changes necessary for adopting robots for service tasks in natural environments is that the robots need to robustly operate in spite of incomplete knowledge of their environment. This necessitates the use of sensors for perception. In addition to perceptual capabilities, robots should also be able to learn from human demonstration. For this learning, embodiment independent representations are necessary as humans and current robotic platforms differ in both perceptual and manipulative skills. A major challenge in this is that the sensors and the embodiments are tightly coupled in sensor-based manipulation. This coupling needs to be decreased to generalize the knowledge. One possible answer to the challenge is to abstract the knowledge and use this abstraction as the basis for the knowledge transfer.

Service robots must be able to cope with several different use cases and tasks, for example setting up a table or placing groceries into a refrigerator. The challenges these tasks present to a robot are twofold. On one hand, the environment poses challenges including a) complexity of the world, such as different number of objects to be handled, b) uncertainty in world knowledge, for example the knowledge of the objects' physical characteristics, and c) dynamic nature of the environment, that is, there are typically other actors in the environment which need to be taken into account. On the other hand, the knowledge transfer from a human to the robot needs to be solved for a particular robot embodiment. The above description demonstrates that there is simultaneously a need for highly embodiment specific information to cope with the uncertain environment and a need to have the embodiment independence to be able to effectively transfer the knowledge between humans and robot embodiments.

In this paper, we propose a new approach, an abstraction architecture, that aims to solve the issue of how the abstract embodiment independent information is used with the embodiment dependent information to cope with the demands of service robotics. The proposed approach uses a hierarchical approach for the decomposition of manipulation skills, which are focused on grasping in this paper, with the ability to use multiple sensors and sensor types. Individual manipulation skills are represented as finite state automata or finite state machine (FSM), with attributes which are used to adapt each skill to a particular use. Using the hierarchical approach ensures that the ability to function in a maximum number of different use cases (for example, different objects, different environment, different hardware) is possible, as the different levels in the hierarchy can be adapted according to the use case. Failure detection is also considered using the finite state automata. The structure of the architecture is shown in Figure 1. The figure shows how the abstract information is completely separated from the embodiment specific information which facilitates the use of multiple embodiments for abstract actions. Combining both the hardware independence and sensor-based manipulation is one of the highlights of the proposed architecture that has not been demonstrated previously, as the hardware independence can not interfere with the real-time requirement of sensor-based manipulation.

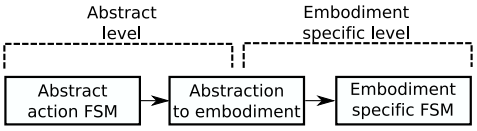


Fig. 1. Levels of the abstraction architecture.

Next, the related work is discussed in Section II. Section III presents the proposed approach. To demonstrate the approach, Section IV shows results of experiments using two different robotic platforms, and Section V concludes the paper with a discussion of the abstraction architecture and future work.

II. RELATED WORK

A number of robot architectures have been presented previously, for both manipulation and for more general

The research leading to these results has received funding from the European Community's Seventh Framework Programme under grant agreement n° 215821, and by Fundació Caixa-Castelló (P1-1A2006-11). V. Kyrki was supported by Academy of Finland grant 114646.

J. Laaksonen and V. Kyrki are with Department of Information Technology, Lappeenranta University of Technology, P.O. Box 20, 53851 Lappeenranta, Finland, jalaakso@lut.fi, kyrki@lut.fi

J. Felip and A. Morales are with Robotic Intelligence Laboratory at the Department of Computer Science and Engineering, Universitat Jaume I, 12006 Castellón, Spain {jfelip,morales}@uji.es

use. Here, we will concentrate on architectures especially targeting manipulation. The concept of primitive skills is central in many of the works as is the use of discrete states to divide a manipulation action into parts.

Milighetti et al. [1] presented an architecture which uses primitive skills, that combine to form a skill, which in turn form a complete task. Each primitive skill is selected by heuristic selection out of many possible primitive skills, based on the sensor signals. A neural network is used to detect the change between the skills. Each primitive skill is based on a separate controller. While the basic idea of hierarchical decomposition is similar to ours, in their approach there is no possibility to adapt the primitive skills themselves.

Haidacher et al. [2] demonstrated an architecture for the DLR Hand II. The architecture is based on different levels of complexity, which handle different aspects of the control. Again, the concept of hierarchical decomposition is central, but the architecture is limited to a single hand and the adaptiveness of the architecture has to be implemented at the highest level as the lower levels are statically defined.

Han et al. [3] present a control architecture for multi-fingered manipulation. As previously, the architecture is based on different levels that handle control from planning to actual joint control. The problem with the architecture is the lack of adaptation as the architecture shows that only predetermined architectural components, such as low level controllers, are available to use. In addition, the architecture does not consider the robotic arm, only the hand.

Hybrid discrete-continuous control architectures for manipulation, such as [4] and [5], separate the control phases according to the state of the manipulator. This is achieved by using discrete events to classify the manipulation configuration and using continuous states to control the dynamic behavior in different configurations. This type of architecture is suitable for both low-level control [5] and for a complete control architecture [4]. Petersson et al. [4] demonstrate a control architecture for a mobile manipulator based on behaviors. The actual manipulator behavior is modelled as a sequence of configurable primitive actions. These primitive actions can be freely defined. These primitives can be chained together using a hybrid automaton to form an action. Although the architecture has some elements desired from a service robotics architecture, such as hardware independence, it lacks the sensor-based approach required to cope in uncertain environments, for example there is no mention of failure detection using the available sensors.

Another mobile manipulator architecture by Chang and Fu [6], is also based on hybrid discrete-continuous control architecture. However, the architecture is more limited than in [4], only consisting of pre-determined set of states, which control the manipulator. These states can be configured for different manipulation tasks. Aramaki et al. [7] have also used automata to control a humanoid robot at a low level.

For a static manipulator, Prats et al. [8] presented a comprehensive system for controlling manipulation. The system also uses automata to control the progress of actions, by separating the primitive actions into the states of the automata. One of the defining features of the architecture is that each state of the automata can be a primitive action or an automaton. This feature can be used to create complex actions. However, the problem of hardware independence is not discussed.

Most of the described architectures have one common element, the use of automata in determining the current state of the control. This approach is also used as part of our proposed approach. However, none of the reviewed architectures describe or demonstrate methods for achieving hardware independence, which is one of the central claims of our approach.

III. ABSTRACTION ARCHITECTURE

Before going in to the details of the proposed approach, we define the terminology used. We use the term *abstraction architecture* for the whole approach. This should not be confused with the control architecture, which is a part of the abstraction architecture related to the actual execution of the actions. Figure 2 shows a general hierarchical decomposition of planning and control. The hierarchy consists of three levels: task, action and primitive. *Task* is the highest level of abstraction, representing a semantically meaningful task such as for example emptying a shopping bag. The task comprises of a sequence of *actions*, which represent subtasks, such as moving an object from one location to another. Actions consist of *primitives*, or primitive actions, which are the lowest level of control in the proposed architecture. More accurate definition for a primitive action used in the proposed architecture is that each primitive action is implemented using a single low-level controller, which is responsible for the actual control of robot hardware.

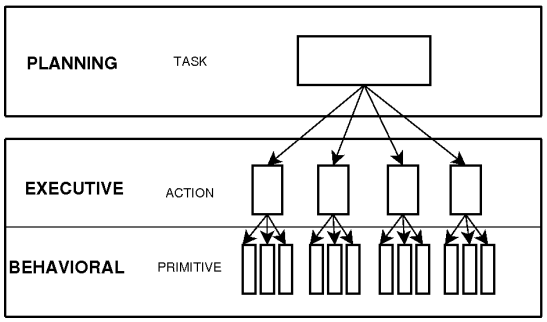


Fig. 2. Planning and control levels.

The abstraction architecture presented in this paper will focus only on the action and primitive levels shown in Figure 2, that is, the actual on-line part of control instead of task planning which might be performed off-line. The architecture itself has elements of both behavioral and executive levels discussed in [9] by Kortenkamp and Simmons. It should be noted that the behavioral control is not considered in the Brooksian sense, instead the behavioral level considers primitive actions which can be executed with traditional control theory. We will show that it is possible to adapt to different tasks and different hardware on the two lower levels

using a set of attributes that are implemented in the actions and in the primitive actions. The focus of the abstraction architecture is on manipulation, especially grasping, by a robotic arm and a robotic hand. Grasping is also used as an example throughout the description of the abstraction architecture.

The control architecture presented in this paper is based on a high level architecture design, which defines the internal structure of the control architecture and the interfaces for hardware and the communication between the controller and outside components. The control architecture itself is not novel, as we have adapted the same idea of using automata for control, seen in Section II. However, the abstraction architecture, i.e., how to combine the abstract actions with the control architecture is novel, and described in detail in sections III-A and III-C. The high level design of the control architecture is depicted in Figure 3 and the actual implementation is detailed in III-B.

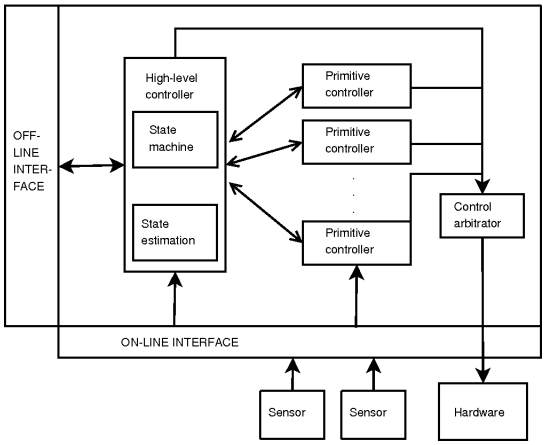


Fig. 3. Control architecture design.

Main features of the control architecture design are the inclusion of two communication interfaces and separation of the high level controller and the primitive controllers. The communication interfaces are for asynchronous ”slow” communication and for real-time communication with sensors and actuators. The high level controller handles the internal state of the controller while the primitive controllers output the control signals to the hardware actuator. The primitive controllers can be freely defined. Final component in the design is the control arbitrator which ensures that a single control input from multiple primitive controllers is communicated to the manipulator.

A. Abstract State Machine

The abstract state machine is a hardware independent description of a manipulation action. The abstract state machine uses XML (eXtensible Markup Language) to describe all relevant information, such as the states and transitions of the state machine. Also information about a target object, e.g. pose and mass, and obstacles in the manipulation environment are given through the XML. All properties and definitions in XML are hardware independent.

TABLE I
STATE AND TRANSITION PROPERTIES.

state	transition
movement	success
hand_shape	grasp_stable
trajectory	grasp_lost
	finger_contact
	finger_contact_lost
	timeout
	collision
	hardware_failure

The abstract state machine is described through definition of states and transitions between the states. Current set of properties for both states and transitions are listed in Table I. The transition properties describe the condition when the transition is triggered. While most of the transition properties are self-explanatory, *success* transition denotes that the controller has reached its target, the state properties are: *movement* describing whether the motion of the manipulator is guarded or free, *hand shape* describing the hand shape with abstract concepts, such as closed or open, and finally, *trajectory* describing a trajectory for the manipulator end-effector, using both position and pose definitions.

In addition to the properties, the state also has attributes, which infer the manipulator motion that is desired from each defined state. These attributes are:

- **success:** The success end state of the state machine.
- **failure:** The failure end state of the state machine.
- **move:** Moving the manipulator without an object.
- **transport:** Moving the manipulator with an object.
- **grasp:** Grasp the object.
- **release:** Release the object.

These attributes are designed with grasping in mind, but other forms of manipulation, such as pushing, are possible to define using the abstract state machine. These attributes are the key factor in selecting the primitive controllers during the translation process, which is described in Section III-C. An example XML definition describing a simple grasp and lift manipulation is shown in Table II. Some of the elements have been left out for brevity, e.g. properties of the object and some of the common transitions, e.g. timeout to the failure state.

B. Embodiment Specific State Machine

The embodiment specific state machine is the functional representation of the abstract state machine. The embodiment specific state machine is able to control the manipulator throughout a single action and decide whether the action was successful or a failure.

The embodiment specific state machine follows the structure of the abstract state machine and the high level design discussed earlier. The high level controller presented in the high level design acts as the single most important element in the proposed control architecture. The high level controller consists of the actual embodiment specific state machine, interfaces to the hardware manipulator, i.e., the robotic arm

TABLE II

LISTING OF THE XML ABSTRACT STATE MACHINE.

```
<statemachine>
 <state name="approach" type="move">
  <movement>free</movement>
  <hand_shape>open</hand_shape>
 </state>
 <state name="preshape_hand" type="move">
  <movement>guarded</movement>
  <hand_shape>pinch_grasp_preshape</hand_shape>
 </state>
 <state name="grasp_object" type="grasp">
  <movement>guarded</movement>
  <hand_shape>pinch_grasp</hand_shape>
 </state>
 <state name="lift_object" type="transport">
  <movement>guarded</movement>
  <hand_shape>pinch_grasp</hand_shape>
  <trajectory>
   <position>0.2 0.6 0.25</position>
  </trajectory>
 </state>
 <state name="success_end" type="success">
 </state>
 <state name="fail_end" type="failure">
 </state>

 <transition origin="approach"
    destination="preshape_hand">
  <success/>
 </transition>
 <transition origin="preshape_hand"
    destination="grasp_object">
  <success/>
 </transition>
 <transition origin="grasp_object"
    destination="lift_object">
  <success/>
  <grasp_stable/>
 </transition>
 <transition origin="lift_object"
    destination="fail_end">
  <grasp_lost/>
 </transition>
 <transition origin="lift_object"
    destination="success_end">
  <success/>
  <grasp_stable/>
 </transition>
</statemachine>
```

and the hand and the control arbitrator. Hardware interface is defined to have one unified control method, Cartesian velocity control, for all arms. However, it is possible to define more control methods for both the arm and the hand of the manipulator. The embodiment specific state machine is modelled as a hybrid discrete-continuous automaton, which was proven successful in many of the reviewed architectures [4], [5], [8]. A hybrid discrete-continuous automaton can also mimic human grasping [10], [11], which consists of several sub-actions.

Each state of the automaton has its own primitive controllers and transitions to other states. As the high level design states, the primitive controllers and transitions are freely definable. However, a common interface for primitive controllers and transitions is required. For primitive controllers the common interface is the control output from the controller and for transitions it is the boolean indication whether the transition to another state should be made or not. Another common interface to both primitive controllers and transitions is setting of attributes which means that we can adapt both the controllers and transitions through these interfaces during the execution of the automaton. All transitions and primitive controllers have also access to all the sensors in the system. Sensor access has been implemented through OpenRAVE [12] and the whole high level controller has been integrated into OpenRAVE, which is an open framework for simulating and controlling robots. OpenRAVE is also used as the off-line interface.

While the embodiment specific state machine is designed to closely resemble the abstract state machine to ease the process of translation between them, it is possible to define the embodiment specific state machine manually to suit needs that can not be described by an abstract state machine. It is also possible to create new states with existing primitive controllers and transitions during the execution of the automaton which enables the use of probabilistic methods such as [13].

C. Translation

The translation process is what combines the abstract state machine and the embodiment specific state machine. The translation takes the abstract state machine as an input, and translates the abstract state machine into an embodiment specific state machine. The translation process is depicted in Figure 4.

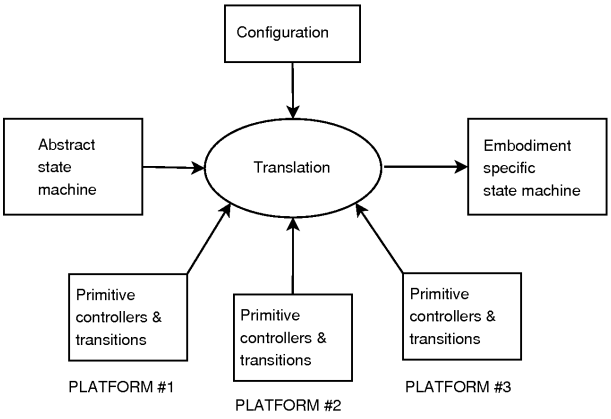


Fig. 4. Translation process.

As can be seen from Figure 4, the translation component needs input defining the configuration of the translation process, i.e. the target platform and the platform specific transitions and primitive controllers used directly in the embodiment specific state machine. The benefit of this arrangement is that the only hardware dependent blocks shown in the figure are the primitive controllers and transitions that are platform specific. Also the critical requirement of real-time operation for sensor-based control is fulfilled as the embodiment specific state machine can be run as is, without any additional overhead from maintaining hardware independence.

The translation process also requires a mapping component which produces the embodiment specific state machine

from the abstract automaton. Currently the mapping itself is done manually per platform, but once the mapping is complete, the translation process from any abstract automaton is performed automatically. This mapping is fairly simple to implement as there are only a limited amount of input properties and the mapping is not aware of the abstract action in any way.

Furthermore, as we have defined a common Cartesian control interface for the arm, we can use primitive controllers that use the arm velocity control for all hardware platforms without modifications. The same applies to some transition conditions, e.g. timeout can be used in all platforms. Thus, building the basic primitive controllers and transitions gives the added benefit of not having to implement all controllers and transitions for each new platform introduced to the system.

IV. DEMONSTRATIONS

The abstraction architecture is demonstrated on two platforms which differ in their kinematics, control, and sensory capabilities. The first platform is a Melfa RV-3SB robot arm with a Schunk PG70 parallel jaw gripper. The arm has 6 DOF (degrees of freedom) and the gripper 1 DOF. In addition to these, a Weiss tactile (pressure) sensor grid is attached to each finger of the gripper. Grasping force is controlled by the feedback from the tactile sensors. Also the stability of the grasp is determined from the tactile sensor feedback.

The second platform consists of a Mitsubishi PA-10 arm with 7 DOF mounted on an Active Media PowerBot mobile robot. The manipulator is endowed with a three-fingered Barrett Hand and a JR3 force/torque and acceleration sensor mounted on the wrist, between the hand and the end-effector. The hand has been improved by adding on the fingertips arrays of Weiss tactile sensors. Each finger of the hand has a built-in strain sensor. The JR3 is a 12 DOF sensor that measures force, torque and acceleration in each direction of the space. The finger-force sensors are used to stop closing the fingers when the object is touched. This sensor is also used to control the force that each finger applies to the object in the final grasp. A complex control grasp primitives that make use of sensor feedback to correct the grasp contacts have been implemented and used for this platform[14].

An executed action is depicted in Figure 5, which shows snapshots of the action being executed on both platforms. The abstract action contains 7 primitive actions: approach, preshape, grasp, lift and move, move down, release and move away. The executions are shown in full for both platforms in the accompanying video. The two objects used in the demonstrations are normal household items, a detergent bottle and a salt container. Both objects have the same mass, 0.5 kg. These objects are shown in Figure 6. In addition to the objects shown, the sensor-based grasping has been demonstrated in the systems with several other similar household objects.

Fig. 6. Grasped objects.

A. Demonstrating Sensor-based Grasping

One of the key challenges that our abstraction architecture addresses is how to combine the need to have sensor based information which is highly coupled to embodiment and abstraction of the action. To show that our abstraction architecture is able to cope with this problem, we demonstrate sensor-based grasping of objects on the two platforms described before. Using the same abstract instructions, i.e., the abstract state machine, we were able to complete the same task on the two platforms, and use the embodiment specific sensors to grasp the object.

As shown in Figure 5 and in the accompanying video, we were able to grasp objects based only on the sensor data from the hand and the arm, no vision was used. Using the same abstract state machine for both platforms shows clearly that we are able to use abstraction and then turn this abstract information to platform specific primitives and transitions used in the sensor-based control.

In the context of the demonstration we were able to use the same controllers for both arms, but the abilities of the hands are too different, in terms of kinematics and sensors, so that both hands had their own controllers. Also the transitions for grasp stability or instability are customized for each of the platforms in order to use the different sensors on the platforms.

B. Failure Detection

Failure detection is an important factor in the proposed architecture. Failure detection can be used to arise surprise and for learning. As the control architecture is focused towards sensor-based control, all the available sensors can be used for failure detection. Failure is also explicitly included in the abstract state machine as one of the end states.

To demonstrate failure detection, the same abstract state machine was used as before with the demonstration platforms. However, the object mass was artificially increased, but this was not reflected on the abstract state machine. Sensor use is critical in detecting failures which can be seen in Figure 7, which shows the result of the demonstration on the first platform. The figure depicts the total force affecting

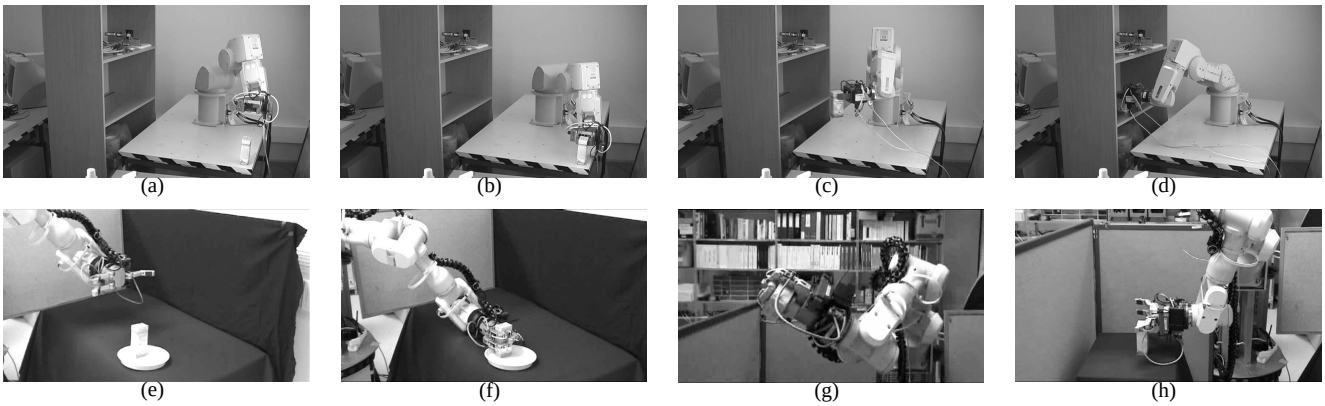


Fig. 5. Action execution on both platforms: (a) Approach; (b) Grasp; (c) Move; (d) Release; (e) Approach; (f) Grasp; (g) Move; (h) Release.

the tactile sensors and the state changes of the state machine as vertical lines. As can be seen from the figure, the state machine was executed normally until the lift and move primitive failed, and the state machine moved into failure state, halting the execution of the state machine. The failure mode is also shown in the accompanying video for one of the platforms. However, both platform have the capability of failure detection, using their platform specific sensors.

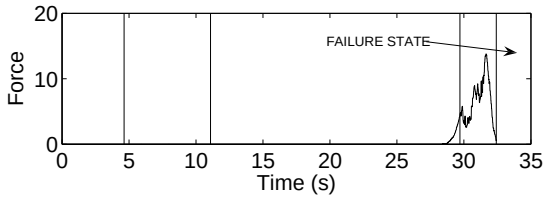


Fig. 7. Measured force during a failed action.

V. DISCUSSION AND CONCLUSIONS

This paper presented an approach for handling embodiment independent knowledge and transferring that knowledge to a more embodiment specific representation which can be used to control the embodiment. Our approach specifically addresses embodiment independence, the use of sensors as an integral part of control, and the modelling of actions as automata of adaptive primitive actions. The embodiment independent knowledge is modelled as a state machine which is then translated to suit each embodiment and its external and internal sensors.

A noteworthy observation is that the use of primitive attributes reduces the problem of learning motions to the learning of primitive attributes. While learning was not demonstrated in this paper, the abstraction would be useful for both imitation learning from a human demonstrator as well as learning by exploration by the robot platform itself. The use of primitives sidesteps the problem of decomposing a trajectory learned from a human to a set of primitives. On the other hand, a known set of primitives must be created and configured for a hand. However by using adaptive primitives, we believe that a wide range of natural motions can be mapped to a limited set of primitives since typical human manipulation is known to consist of a limited number of types of interaction.

Our future work includes implementing the abstraction architecture on more platforms. The abstraction architecture will then form the base for further research on the use of sensors as a part of manipulation and especially as a part of manipulation learning. The focus will be on grasping and how to learn to grasp using the available sensors in both real and simulated environments.

REFERENCES

[1] G. Milighetti, H. B. Kuntze, C. W. Frey, B. Diestel-Feddersen, and J. Balzer, "On a primitive skill-based supervisory robot control architecture," in *Proc. IEEE ICRA'05*, July 2005, pp. 141–147.

[2] S. Haidacher, J. Butterfass, M. Fischer, M. Grebenstein, K. Joehl, K. Kunze, M. Nickl, N. Seitz, and G. Hirzinger, "DLR hand II: Hard- and software architecture for information processing," in *Proc. IEEE ICRA'03*, Sept. 2003, pp. 684–689.

[3] L. Han, Z. Li, J. C. Trinkle, Z. Qin, and S. Jiang, "The planning and control of robot dextrous manipulation," in *Proc. IEEE ICRA'00*, Sept. 2000, pp. 263–269.

[4] L. Petersson, M. Egerstedtt, and H. Christensen, "A hybrid control architecture for mobile manipulation," in *Proc. IEEE/RSJ IROS'99*, 1999, pp. 1285–1291.

[5] T. Schlegl and M. Buss, "A discrete-continuous control architecture for dextrous manipulation," in *Proc. IEEE SMC'99*, vol. 2, 1999, pp. 860–865.

[6] C.-F. Chang and L.-C. Fu, "A hybrid system design of a mobile manipulator," in *Proc. IEEE ICRA'06*, May 2006, pp. 406–411.

[7] S. Aramaki, H. Shirouzu, and K. Kurashige, "Control program structure of humanoid robot," *IECON 02*, vol. 3, pp. 1796–1800 vol.3, Nov. 2002.

[8] M. Prats, A. P. del Pobil, and P. J. Sanz, "A control architecture for compliant execution of manipulation tasks," in *Proc. IEEE/RSJ IROS'06*, Oct. 2006, pp. 4472–4477.

[9] D. Kortenkamp and R. Simmons, "Robotic systems architecture and programming," in *Springer Handbook of Robotics*, 1st ed., B. Siciliano and O. Khatib, Eds. Berlin, Germany: Springer-Verlag, 2008.

[10] U. Castiello, "The neuroscience of grasping," *Nature Neuroscience*, vol. 6, pp. 726–736, Sep 2005.

[11] I. Serrano Vicente, V. Kyrki, D. Kragic, and M. Larsson, "Action recognition and understanding through motor primitives," *Advanced Robotics*, vol. 21, no. 15, pp. 1687–1707, 2007.

[12] R. Diankov and J. Kuffner, "OpenRAVE: A planning architecture for autonomous robotics," Robotics Institute, Carnegie Mellon University, Tech. Rep. CMU-RI-TR-08-34, July 2008.

[13] K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez, "Grasping POMDPs," in *Proc. IEEE ICRA'07*, Apr. 2007, pp. 4685–4692.

[14] J. Felip and A. Morales, "Robust sensor-based grasp primitive for a three-finger robot hand," in *IEEE/RSJ International. Conference on Intelligent Robots and Systems*, Oct. 2009.

Robust sensor-based grasp primitive for a three-finger robot hand

Javier Felip and Antonio Morales

Abstract— **This paper addresses the problem of robot grasping in conditions of uncertainty. We propose a grasp controller that deals robustly with this uncertainty using feedback from different contact-based sensors. This controller assumes a description of grasp consisting of a primitive that only determines the initial configuration of the hand and the control law to be used.**

We exhaustively validate the controller by carrying out a large number of tests with different degrees of inaccuracy in the pose of the target objects and by comparing it with results of a naive grasp controller.

I. Introduction

Management of uncertainty is one of the biggest problem to address when developing applications for unstructured scenarios. In the case of robot grasping, uncertainty can arise from several sources: lack of complete knowledge about the physical properties and shape of the target objects, inaccuracy in the determination of the pose of the object and the configuration of the robot (i.e.: position of mobile robot), mismatch between planner models and real conditions due to sensing errors, limitation of planner models, and many others.

Analytical solutions to the grasp planning problem has been provided for structured scenarios [1]. However these solutions often depend on the assumption that the contact locations obtained as solutions are reachable by actuators with enough precision. For common robots scenarios this is not realistic even in the case that the shape of the object is perfectly known. Several attempts to design analytical grasp planning algorithms that take into account a certain degree of inacuracy has been made [2], [3]..

A common approach to reduce uncertainty is the use of sensor information in the planning and execution phases of grasping. Vision has been used to obtain the shape of unknown target objects [4], [5], or to determine the location and pose of them [6]. In both cases, visual input is used to plan feasible grasps. Visual feedback is also used when the arm tries to reach the object. Murphy et al. uses visual techniques to correct the orientation of four-finger hand while approaching an object to allow better contact locations [7]. Namiki et al. uses a fast control schema in combination with tactile feedback to cage an object [8]. Infrared sensors has been also used to correct the approaching orientation of the gripper [9]. In innovative design Hsiao et al. use IR sensors to estimate the normal direction of closer object surfaces to search a suitable contact location [10].

J. Felip and A. Morales are with Robotic Intelligence Laboratory at the Department of Computer Science and Engineering, Universitat Jaume I, 12006 Castellón, Spain {jfelip,morales}@uji.es

Fig. 1. PA-10 7 d.o.f with Barrett Hand and a JR3 force/torque and acceleration sensor. The hand has Weiss Robotics pressure sensors on its fingertips.

Once the object is contacted with the robot, gripper, tactile and force sensors can be applied. Contact force measurement is used to estimate the quality of the grasps [11], [12], [13] or the shape of the object [14] with the purpose of reach better contact locations through a sequence of grasping/regrasping actions. Contact information can also be used to program complex dexterous manipulation operations like finger repositioning while holding the object [11], [15]. Several works have combined the use of several sensors to complete the whole process of grasp planning and execution [16], [17].

Robustness in grasp execution is not only achieved by designing sensor-based controllers but also by combining several controllers with different optimisation goals. These combinations has been based on hierarchical schemes based on reflex programming [18], [13] or complementary controllers [12], [10].

A. Grasp primitives

In this paper we follow a sensor-based approach that is based on an alternative paradigm of describing grasps. Most of the above papers assume that grasps are described as a set of contact points on the object surface. In fact, most of the problems arise as a consequence of the impossibility of reaching those points. Our paper is developed under a different assumption. Grasps are described as instances of basic primitives [17]. A grasp primitive is a specific controller designed to perform a particular indivisible action, in our case a grasp. In practical terms it is defined by a initial hand preshape, a sensor-based controller, and a set of ending conditions. Its behaviour can be determined by several parameters like initial position and orientation, maximum force allowed, and others. An *instance* of a grasp primitive is the set of values of this parameters. Hence, a

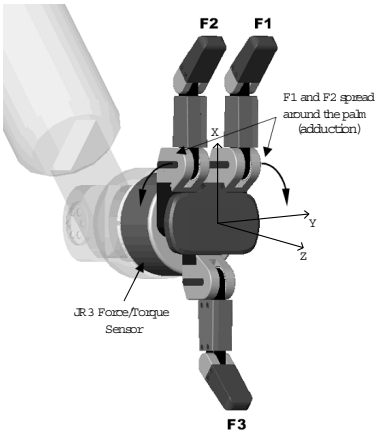


Fig. 2. Detail of the Barrett hand.

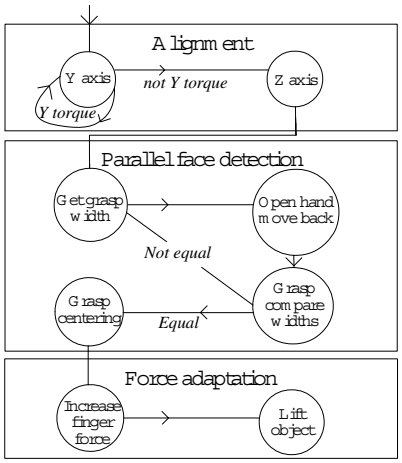


Fig. 3. Algorithm execution diagram.

grasp is an instance of a primitive that determines the initial configuration of the robot hand and the control policy that is going to be applied to execute the grasp.

This definition of grasp primitive presents two perspectives. From a practical point of view a grasp primitive is a single controller that performs a specific task on a particular embodiment. From an abstract point of view, primitives are the simplest pieces of a vocabulary to elaborate plans. Hence, they are well suited to be the basic piece of a reasoning and learning procedures.

The concept of grasp primitive is not new and has been used in many other robot-related works. Actually the term "motor primitive" is borrowed from neuroscience literature [19], and has also been widely used in robot learning [20], [21]. Nagatani and Yuta implemented and combined several action primitives to perform a complex behaviour: a mobile robot behaviour capable of opening and going through a door [22]. Aarno et al. implement visual analysis to program "Elementary Grasping Actions (EGA)", a kind of grasp primitives, for a parallel gripper [5]. Finally Finite State Machines has been proposed to combine primitive actions in the execution of complete manipulation tasks [23].

II. Methodology

A. System description and Assumptions

We implemented our primitive for a robotic setup consisting of a Mitsubishi PA-10 with 7 d.o.f. (Degrees of freedom) mounted on an Active Media PowerBot mobile robot. The manipulator is endowed with a three-fingered Barrett Hand and a JR3 force/torque and acceleration sensor mounted on the wrist, between the hand and the end-effector (see Fig. 1). The hand has been improved by adding on the fingertips arrays of pressure sensors designed and implemented by Weiss Robotics.

The Barrett hand is a 4 d.o.f., three-fingered hand. Each finger has one degree of freedom thus phalanxes are not independent. Fingers F1 and F2 can rotate around the palm and move next to Finger F3 (Thumb) or oppose to it, this d.o.f. is called adduction. The reference frame of the hand and the adduction d.o.f. are depicted in Fig. 2. Each finger of the hand has built-in strain sensor. The JR3 is a 12 d.o.f. sensor that measures force, torque and acceleration in each direction of the space.

Our experimental workspace consists of a horizontal surface where the targets objects are lying. The objects that can be manipulated are those that can fit inside the hand. The minimum dimensions are 25mm height, 70mm long, and 10mm width. Big objects that can be held by the hand should have a maximum width of 200mm. On this paper we have focused on box-like and clyinder-like objects within those dimensions (see Figs. 6, 7 and 8) and have not considered non-symetrical objects.

The input of the primitive controller is the starting position and orientation of the hand and the maximun finger force. In optimal conditions, the hand will be perfectly oriented in the direction of the object, and rotated perpendicularly with respect to the main axis of the object bounding box. These input parameters are provided from an external module based on visual information.

The designed controller is able to deal with errors in determining the parameters. The error estimitation is decomposed in translation error and rotation error (see Fig. 9). The first is defined by the cartesian distance on each frame axis between the center of the object and its estimation. The second is calculated from the difference between each estimated object axis and its true orientation.

The controller tries to grasp the object aproaching from above following an orientation close to the vertical. In order to allow lateral approaching directions several changes in the desing of the contorller may be necessary. Basically an estimation of the distance to the object should be known in advance. During the grasp execution, the robot can inadvertly move the object but the controller is designed to take into account most of these cases.

The algorithm starts with a cylindrical preshape. Where two fingers (F1 and F2) are opposing completely the thumb (F3). In some cases, the hand preshape is switched to a spherical configuration where the fingers are arranged

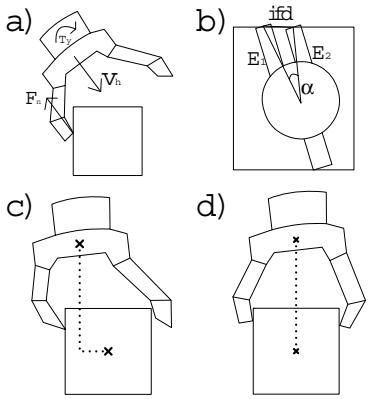


Fig. 4. Correction of alignment errors. a) The hand touches the object. The contact is not perpendicular and a normal force (Fn) appears at a distance from the center creating the torque Ty. b) The hand Z error is calculated using Index and Middle finger extensions. c) The grasp center is displaced.

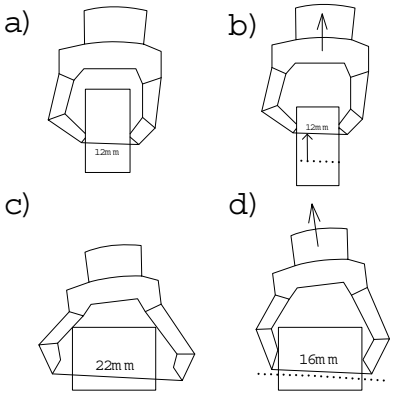


Fig. 5. Determining parallelism of grasped faces: a) First contact. b) Grasp width is constant, the faces grasped are parallel. c) Inner phanlanxes contact the object.

forming an equilateral triangle.

B. Algorithm

The controller tries to obtain grasp stability on the basis of the following criteria (ordered by relevance):

- Hand-object alignment
- Parallelism of contacted faces
- Maximization of contact surface
- Finger position symmetry
- Finger force symmetry

Hand and object are aligned when all their axis are parallel and the projection of the Z axis of the hand intersects the object bounding box on its center. The alignment avoids torque forces from appearing when lifting the object.

The starting position and orientation of the hand is a critical parameter for the algorithm. This pose sets the approach vector to the object which is along the positive Z axis of the hand (see Fig. 2). The aim of the algorithm is to perform a stable grasp of the object allowing a considerable error in the starting position. Thus the success of the grasp is less dependent from the accuracy of the estimations.

The execution of the grasp is divided into three main phases (see Fig. 3).

1) Alignment: This phase tries to align hand and object using force and tactile feedback. First of all the hand moves forward until the force/torque sensor on the wrist detects contact with the object. If this contact causes a torque force around Y axis it means that the object and the hand are not aligned (see Fig. 4.a). To correct this error the hand moves back and rotates an angle of two degrees, then continues touching the object until the torque disappears or it changes the sign. If the torque changes the sign, the hand rotates one degree in the opposite direction and the Y alignment ends.

At this point the hand closes. The difference in the extension of the fingers F1 and F2, determines the rotation around the Z axis (see Fig. 4.b) needed to align with the object. Both fingers must have the same extension to perform an stable grasp. This correction is not applied if the spherical pregrasp shape is set.

2) Parallel face detection: In the second stage the controller tries to determine if the contacted surfaces are parallel and stable. Using the Barrett Hand inner force sensors to stop the fingers and the hand propioception, the width of the current grasp is measured (see Fig. 5.a and c), then the hand opens a little and moves 5mm backwards. After that the hand closes and the width of the grasp is measured again. If there is a big difference between the two samples it means that the grasped faces are not parallel (see Fig. 5.d) and the process starts again. This phase of the algorithm repeats until the difference is close to zero or the object is lost. If the object is lost a reflex to recover it, is triggered. This reaction is explained at the end of this section.

When the grasp is stable(i.e. aligned with object and grasped surfaces parallel), the fingers move the object aligning it with the palm center and keeping the extension of the fingers (see Fig. 4.c and d) in order to improve contact surface, finger position symmetry and finger force symmetry.

3) Force adaptation: Using the fingertip integrated force sensors of the Barrett hand, the force of each fingertip is increased until it reaches the predefined limit. Then, the hand lifts the object and evaluates if the grasp has been successful.

C. Security reflexes

During the execution of the primitive, the algorithm is attentive for some important events in order to inform the user, adapt to the environment conditions and perform a successful grasp. The first event is the loss of the object. This happens when all the three fingers close completely without making any contact on the object. In this case the hand opens and moves a little bit down then closes again trying to recover the object contact.

Another event is the adduction of the fingers. When the fingers make contact, if the object is cylindrical, the fingers can adduct due to the adduction d.o.f is set free. This natural adduction is detected by the algorithm and the hand configuration is set to spherical.

The last event is the miss of the object by only one finger. The reaction is to open the hand and to move laterally 5cm (inter finger distance).

Fig. 6. Cylinder-like objects. Properties (radius x height, weight) from left to right and top to bottom: Cylinder1(110x80, light) Cylinder2(65x215, light) Cylinder3(105x75, heavy) Cylinder4(115x50, light)

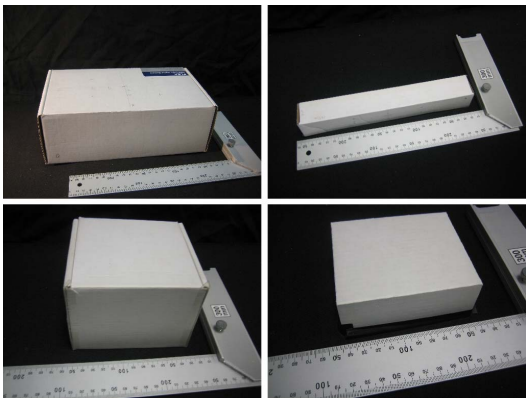


Fig. 7. Box-like objects. Properties (base x height, weight) from left to right and top to bottom: Box1(270x53x95, light) Box2(236x35x35, light) Box3(127x116x92, heavy) Box4(100x87x45, light)

D. Additional parameters

Other parameters as distance, size, weight and shape could be used to improve the accuracy and execution of the grasp. The distance could be used to avoid blind first contact with the object; the size could be used to set the starting opening of the fingers; the weight to determine the force to be applied by the fingers; and the shape to set the pregrasp shape reducing the time consumed by the pregasp shape detection and switching.

III. Validation

A test bench has been designed in order to validate the grasping controller. This test bench consists of a set of objects and a set of starting positions to be tried with each object.

To have a comparison reference for our controller, we have designed an alternative naive grasp controller without corrections. This controller needs 4 input parameters: starting position, distance to the object, pregrasp size and finger force. The fingers moves to the pregrasp size and the hand moves forward along its Z axis the distance specified. The hand closes and lifts the object. If the object is lifted and does not fall for 10 seconds, the execution is successful.

A. Objects and test bench

The objects selected are classified according to their shape (cylinders in Fig. 6, boxes in Fig. 7 and others in Fig. 8), their size (thin, normal, thick) or their weight (light, heavy). All the objects are solid. Following the shape classification, we have selected thin, normal and thick objects for each shape in order to test as many different combinations of object features as possible. The optimal conditions have been tested in all the objects. We have selected a subset of 2 box-like objects and 2 cylinder-like objects to test rotation and translation error conditions. This selected objects are the biggest and the smallest from each category.

Translation error is the deviation from the center of the object to the center of the hand and it is measured in mm. Rotation error is the deviation between hand and object main axis and is measured in degrees (see Fig. 9).

This test bench evaluates robustness against translation and rotation errors. The behaviour of each algorithm has been also evaluated in optimal conditions which can present a rotation error of 5 degrees and 10% of translation error.

To evaluate the effect of rotation errors the following conditions have been taken into consideration:

- 15 degrees on X, Y, Z, XY, XZ, YZ and XYZ.
- 20 degrees on X, Y, Z

The combined rotation error is applied first in X next in Y and later in Z. A rotation error of 15 degrees in XYZ is a rotation of 15 degree on X, then 15 degree on Y and finally 15 degree on Z. The results of the rotation tests are shown in Table III for the robust controller and in Table IV for the naive controller. The cylinder-like objects are invariant to rotation in Z axis. The Z rotation error is not applicable to cylinder-like objects.

The amount of translation error is relative to the size of the object because usually this two variables (size and error) are related. To evaluate the effect of translation errors the following conditions have been taken into consideration:

- 20% on X, Y, Z, XY, XZ, YZ and XYZ
- 40% on X, Y, Z

IV. Results

The global results are presented in Table I and Table II, the first column shows the results for the optimal case, the

Fig. 8. Other objects. Properties (base x height, weight) from left to right: Other1(180x90x90, heavy) Other2(90x90x163, light)

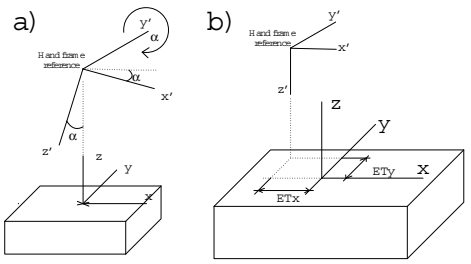


Fig. 9. Rotation and translation error, the object frame reference is on the center of the object. a) Example of alpha degrees Y rotation error. b) Example of X and Y translation error

	Optimal	Rotation	Translation	Total
Box 1	5/5(100%)	24/24(100%)	13/24(54%)	42/53(79%)
Box 2	5/5(100%)	16/22(73%)	22/24(92%)	43/51(84%)
Cylinder 2	5/5(100%)	7/12(58%)	16/20(80%)	28/37(76%)
Cylinder 4	5/5(100%)	11/11(100%)	19/20(95%)	35/36(97%)

TABLE I

ROBUST ALGORITHM GLOBAL RESULTS

second and third columns present the summary of the rotation and translation error. The last column shows the averaged results for each object.

Details about experiments with error conditions can be found in Table III and Table IV for rotation error and in Table V and Table VI for translation error.

V. DISCUSSION

Summary tables I and II show clearly the better performance obtained by our robust controller in comparison with the naive one. It not only successes in a 100% of the experiments in optimal conditions but also outperforms

	Optimal	Rotation	Translation	Total
Box 1	5/5(100%)	25/50(50%)	16/40(40%)	46/95(48%)
Box 2	5/5(100%)	26/50(52%)	40/40(100%)	71/95(75%)
Cylinder 2	5/5(100%)	23/25(92%)	35/40(88%)	63/70(90%)
Cylinder 4	5/5(100%)	24/25(96%)	14/20(70%)	43/50(86%)

TABLE II

NAIVE ALGORITHM GLOBAL RESULTS

Error	Box 1	Box 2	Cylinder 2	Cylinder 4
15 X Axis	2/2(100%)	2/2(100%)	2/2(100%)	2/2(100%)
20 X Axis	2/2(100%)	2/2(100%)	2/2(100%)	2/2(100%)
15 Y Axis	3/3(100%)	2/2(100%)	1/2(50%)	3/3(100%)
20 Y Axis	3/3(100%)	2/2(100%)	1/4(25%)	2/2(100%)
15 Z Axis	3/3(100%)	2/2(100%)	N/A	N/A
20 Z Axis	3/3(100%)	2/2(100%)	N/A	N/A
15 XY Axis	2/2(100%)	1/2(50%)	2/2(100%)	2/2(100%)
15 XZ Axis	2/2(100%)	1/2(50%)	N/A	N/A
15 YZ Axis	2/2(100%)	1/2(50%)	N/A	N/A
15 XYZ Axis	2/2(100%)	1/4(25%)	N/A	N/A

TABLE III

RESULTS WITH ROTATION ERROR FOR THE ROBUST ALGORITHM

Error	Box 1	Box 2	Cylinder 2	Cylinder 4
15 X Axis	5/5(100%)	5/5(100%)	5/5(100%)	5/5(100%)
20 X Axis	5/5(100%)	5/5(100%)	5/5(100%)	5/5(100%)
15 Y Axis	4/5(80%)	2/5(20%)	3/5(20%)	4/5(80%)
20 Y Axis	2/5(20%)	0/5(0%)	5/5(100%)	5/5(100%)
15 Z Axis	4/5(80%)	5/5(100%)	N/A	N/A
20 Z Axis	4/5(80%)	5/5(100%)	N/A	N/A
15 XY Axis	0/5(0%)	0/5(0%)	5/5(100%)	5/5(100%)
15 XZ Axis	0/5(0%)	4/5(80%)	N/A	N/A
15 YZ Axis	1/5(20%)	0/5(0%)	N/A	N/A
15 XYZ Axis	0/5(0%)	0/5(0%)	N/A	N/A

TABLE IV

RESULTS WITH ROTATION ERROR FOR THE NAIVE ALGORITHM

Error	Box 1	Box 2	Cylinder 2	Cylinder 4
20% X Axis	2/2(100%)	2/2(100%)	2/2(100%)	2/2(100%)
40% X Axis	1/2(50%)	2/2(100%)	2/2(100%)	2/2(100%)
20% Y Axis	1/2(50%)	2/2(100%)	1/2(50%)	2/2(100%)
40% Y Axis	0/2(0%)	2/2(100%)	0/2(0%)	1/2(50%)
20% Z Axis	2/2(100%)	2/2(100%)	2/2(100%)	2/2(100%)
40% Z Axis	2/2(100%)	2/2(100%)	2/2(100%)	2/2(100%)
20% XY Axis	1/4(25%)	3/4(75%)	2/2(100%)	2/2(100%)
20% XZ Axis	2/2(100%)	1/2(50%)	2/2(100%)	2/2(100%)
20% YZ Axis	1/2(50%)	1/2(50%)	1/2(50%)	2/2(100%)
20% XYZ Axis	1/4(25%)	3/4(75%)	2/2(100%)	2/2(100%)

TABLE V

RESULTS WITH TRANSLATION ERROR FOR THE ROBUST ALGORITHM

the naive one when rotational and translational errors are introduced.

The only exemption to this rule is the case of Cylinder 2 (object on the top-left corner on fig 6). This object is too light and when touched while lying on a surface, it moves easily. We observed that the successive contacts that our controller produce causes that the object variates its position making impossible to grasp it.

This case shows one of the drawbacks of our approach. Our controller is touching the objects several times before finally closing the finger to catch them. In case of light or unstable objects this can be a problem. This difficultty surpassed by the use of more sensitive sensors or by the implmentation of compliant hardware or controllers. The use of proximitiy sensors [10] would completely solve this

Error	Box 1	Box 2	Cylinder 2	Cylinder 4
20% X Axis	0/4(0%)	4/4(100%)	4/4(100%)	4/4(100%)
40% X Axis	0/4(0%)	4/4(100%)	3/4(75%)	4/4(100%)
20% Y Axis	4/4(100%)	4/4(100%)	4/4(100%)	4/4(100%)
40% Y Axis	0/4(0%)	4/4(100%)	4/4(100%)	0/4(0%)
20% Z Axis	3/4(75%)	4/4(100%)	4/4(100%)	2/4(50%)
40% Z Axis	2/4(50%)	4/4(100%)	2/4(50%)	2/4(50%)
20% XY Axis	0/4(0%)	4/4(100%)	4/4(100%)	4/4(100%)
20% XZ Axis	2/4(50%)	4/4(100%)	4/4(100%)	2/4(50%)
20% YZ Axis	4/4(100%)	4/4(100%)	2/4(50%)	4/4(100%)
20% XYZ Axis	1/4(25%)	4/4(100%)	4/4(100%)	2/4(50%)

TABLE VI

RESULTS WITH TRANSLATION ERROR FOR THE NAIVE ALGORITHM

problem.

One of the advantages of our approach is the little previous information it needs about the object. No exact model of the object is necessary, and the only input is the maximum force to be applied by the fingers. It is supposed that the hand is appropriately oriented and preshaped. More information, like estimated size or the distance, would be definitively help to improve the controller robustness and the time necessary to complete a grasp.

Currently all the grasp tried to approach from above. That is, the objects are lying on a surface and the hand approaches them vertically. This simplifies our controller since the movements of the objects are limited. Improvements are necessary if grasps from a side are going to be executed, since the stability of the objects could be compromised if they are touch. In this case an estimation of the distance to the object would be necessary.

At the moment the average time to execute a grasp is about 40 seconds, though this time depends on the object and the initial position error. It could be reduced providing more information about the location and characteristics of the objects.

Finally, an attached video shows pose correction phases, event adaptation and grasp force increasing. It is also shown that the stability of the grasps performed by the robust algorithm are better than the ones performed by the naive controller.

VI. Conclusion

We have developed a robust sensor-based grasp primitive that need little information to execute its task and that is able the correct and adapt to variations and inaccuracies in the expected conditions of the scenario.

We indicated three ways of improving the grasp primitive controller implemented. The most immediate future work is to extend the family of manipualtion primitives that allow the execution of a complete pick-and-place task, i.e.: approaching and preshaping, lifting, transportation and landing primitives. The development of these primitives would provide a vocabulary of basic skills that will allow planning, and learning in future stages.

Acknowledgment

This paper describes research carried out at the Robotic Intelligence Laboratory of Universitat Jaume I. The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement 215821 (GRASP project), and by Fundació Caixa-Castelló (P1-1A2006-11).

References

[1] A. Bicchi, "Hand for dexterous manipulation and robust grasping: A difficult road towards simplicity," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 6, pp. 652–662, 2000.

[2] R. C. Brost, "Planning robot grasping motions in the presence of uncertainty," Tech. Rep. CMU-RI-TR-85-12, Computer Science Department and Robotics Institute, Carnegie Mellon University, 1985.

[3] Y. Zheng and W.-H. Qian, "Coping with the grasping uncertainties in force-closure analysis," *International Journal of Robotics Research*, vol. 24, no. 4, pp. 311–327, 2005.

[4] A. Morales, P. Sanz, A. del Pobil, and A. Fagg, "Vision-based three-finger grasp synthesis constrained by hand geometry," *Robotics and Autonomus Systems*, vol. 54, pp. 496–512, June 2006.

[5] D. Aarno, J. Sommerfeld, D. Kragic, N. Pugeault, S. Kalkan, F. Wörgötter, D. Kraft, and N. Krüger, "Early reactive grasping with second order 3D feature relations," in *IEEE Conference on Robotics and Automation (submitted*, 2007.

[6] P. Azad, T. Asfour, and R. Dillmann, "Combining appearance-based and model-based methods for real-time object recognition and 6D-localization," in *International Conference on Intelligent Robots and Systems (IROS)*, (Beijing, China), 2006.

[7] T. Murphy, D. Lyons, and A. Hendriks, "Stable grasping with a multi-fingered robot hand: A behavior-based approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, (Yokohama, Japan), pp. 867–874, July 1993.

[8] A. Namiki and M. Ishikawa, "Optimal grasping using visual and tactile feedback," in *IEEE/SICE/RSJ Intl. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, (Whashington, DC), pp. 589–596, Dec. 1996.

[9] M. Teichmann and B. Mishra, "Reactive robotics I: Reactive grasping with a modified gripper and multi-fingererd hands," *International Journal of Robotics Research*, vol. 19, no. 7, pp. 697–708, 2000.

[10] K. Hsiao, P. Nangeroni, M. Huber, A. Saxena, and A. Y. Ng, "Reactive grasping using optical proximity sensors," in *IEEE International Conference on Robotics and Automation*, (Kobe, Japan), May 2009. To appear.

[11] J. Coelho Jr. and R. Grupen, "A Control Basis for Learning Multifingered Grasps," *Journal of Robotic Systems*, vol. 14, pp. 545–557, Oct. 1997.

[12] R. Platt Jr., A. H. Fagg, and R. Gruppen, "Nullspace composition of control laws for grasping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Lausanne, Switzerland), pp. 1717–1723, 2002.

[13] T. Mouri, H. Kawasaki, and S. Ito, "Unknown object grasping strategy imitating human grasping reflex for anthropomorphic robot hand," *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, vol. 1, no. 1, pp. 1–11, 2007.

[14] P. Allen and K. Roberts, "Haptic object recognition using a multi-fingered dextrous hand," *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pp. 342–347 vol.1, May 1989.

[15] M. Huber and R. Grupen, "Robust finger gaits from closed-loop controllers," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, pp. 1578–1584 vol.2, 2002.

[16] P. Allen, A. T. Miller, P. Oh, and B. Leibowitz, "Using tactile and visual sensing with a robotic hand," in *IEEE International Conference on Robotics and Automation*, (Albuquerque, New Mexico), pp. 677–681, Apr. 1997.

[17] B. J. Grzyb, E. Chinellato, A. Morales, , and A. P. del Pobil, "Robust grasping of 3D objects with stereo vision and tactile feedback," in *International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR)*, (Coimbra, Portugal), pp. 851 – 858, 2008.

[18] K. Imazeki and T. Maeno, "Hierarchical control method for manipulating/grasping tasks using multi-fingered robot hand," *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 4, pp. 3686–3691 vol.3, Oct. 2003.

[19] F. A. Mussa-Ivaldi, S. F. Giszter, and E. Bizzi, "Linear combinations of primitives in vertebrate motor control," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 91, no. 16, pp. 7534–7538, 1994.

[20] M. Ferch and J. Zhang, "Learning cooperative grasping with the graph representation of a state-action space," *Robotics and Autonomous Systems*, vol. 38, pp. 183–195, 2002.

[21] R. Amit and M. J. Mataric, "Parametric primitives for motor representation and control," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA-2002*, pp. 863–868, 2002.

[22] K. Nagatani and S. Yuta, "Door-opening behavior of an autonomous mobile manipulator by sequence of action primitives," *Journal of Robotic Systems*, vol. 13, no. 11, pp. 709–721, 1996.

[23] D. Kragic, S. Crinier, D. Brunn, and H. Christensen, "Vision and tactile sensing for real world tasks," *IEEE International Conference on Robotics and Automation*, pp. 1545–1550, September 2003.

Learning grasp stability based on haptic data

Author Names. Paper-ID 174

Abstract— **Sensor based grasping of objects and grasp stability estimation is an important skill for a general purpose robot. Grasp stability modeling and estimation has been studied for a long time but there are very few robots today that can demonstrate extensive grasping skills. The main contribution of the work presented here is the use of machine learning methods for inferring of grasp stability based on tactile measurements. The main objective is to classify a grasp as stable or unstable before applying further actions on it. The problem is important and cannot be solved by vision sensing typically used as an input to a grasping control loop. The output of the classification system can trigger a re-grasping step if an unstable grasp is identified.**

An off-line learning process is implemented and used for reasoning about grasp stability on a three-fingered robotic hand. We study both one-shot and time series classification methods. To evaluate the proposed method, experiments are performed both in simulation and on a real robot system. The results show that the idea of exploiting the machine learning approach is feasible and it opens a number of interesting venues for the future research.

I. Introduction

Grasping is an essential skill for a general purpose service robot, working in an industrial or home-like environment. The classical work in robotic grasping assumes that the object parameters such as pose, shape, weight and material properties are known. If precise knowledge of these is available, grasp planning using analytical approaches, such as form or force closure, may be enough for successful grasp execution. However, in unstructured environments the information is usually uncertain, which presents a great challenge for the current state-of-the-art work in this area.

Sensors can be used to alleviate the problem of uncertainty. To determine the shape and pose of an object, vision has been commonly used. However, the accuracy of vision is limited and small errors in object pose are frequent even for known objects. It is not uncommon that even these small errors cause failures in grasping. These failures are also difficult to prevent at the grasp planning stage. This problem is magnified when also the object models are acquired on-line using vision or other similar sensors. While the tactile and finger force sensors can be used to reduce this problem, a grasp may fail even when all fingers have adequate contact forces and the hand pose is not dramatically different from the planned one.

The main contribution of this paper is to show that it is possible to infer knowledge about grasp stability using information from tactile sensors while grasping an object before beining further manipulated. This is very useful, because if failures can be detected, objects can be regrasped before trying to lift them. However, the relationship between tactile measurements and grasp stability is embodiment specific and very complex. For this reason, we propose to use machine learning techniques for the inference. To achieve good generalization performance, machine learning approaches typically require large amount of training data. As a solution to the problem of acquiring enough training data, we propose to simulate the grasping process. However, we evaluate the feasibility of the approach both on simulated and real data.

We first study one-shot recognition: detecting grasp stability from a single tactile measurement. We also implement the time-series analysis approaches based on a sequence of tactile measurements. The results show that the idea of exploiting the machine learning approach is feasible and it opens a number of interesting venues for the future research. As an additional contribution of the work presented here is the generation of a database with examples of tactile measurements for stable and unstable grasps on a set of objects. The data will be publically available at http://xxx.yyy.zzz. We believe that the tactile database will be compliment to the Columbia Grasp Database [1].

Next, related work is reviewed in Sec. II. In Sec. III the simulator and the constructed tactile database are described. Section IV presents the one-shot recognition approach using support vector machine classification and results of the approach on simulation data. Then, Sec. V introduces the time-series recognition approach using hidden Markov models and the related experimental results. Finally, we conclude and present directions for future research in Sec. VI.

II. Related work

In robotic object grasping there has been a lot of effort during the past few decades (e.g see [2] for a recent survey). There are some recent examples which base grasp generation on visual input and use tactile sensing for closed loop control once in contact with the object. For example, the use of tactile sensors has been proposed to maximize the contact surface for removing a book from a bookshelf [3]. Application of force, visual and tactile feedback to open a sliding door has been also proposed [4]. In our work the main difference is that the tactile sensors are used to assess the stability of a grasp. Thus, rather than using the tactile data for control, we reason about grasp stability.

Most of the grasp planning approaches tested in simulation have the common property of using a strategy that relies on the object shape. Modelling object shape with a number of primitives such as boxes, cylinders, cones, spheres [5], or superquadrics [6] reduces the space of possible grasps. The decision about the suitable grasp is made based on grasp quality measures given contact positions. However, these kind of techniques do not provide a way of dealing with

uncertainties that might arise in dynamic scenarios which can be solved using tactile feedback.

Learning aspects have been considered in the context of grasping mostly for the purpose of understanding human grasping strategies. In [7], it was demonstrated how a robot system can learn grasping by human demonstration using a grasp experience database. The human grasp was recognized with the help of a magnetic tracking system and mapped to the kinematics of the robot hand using a predefined lookup-table. Current learning approaches using tactile sensors are focused on either determining the properties of objects [8, 9] or object recognition [10, 11].

To our knowledge, analysis of grasp stability using machine learning techniques and tactile sensors has not been demonstrated before.

III. Simulator and the Database

In order for a learning system to be able to generalize to a wide variety of cases, relatively large training data is usually required. Generating large datasets on real hardware is time consuming and in robotic grasping generating repeatable experiments is difficult due to the dynamics of the grasping actions. Generating thousands of samples is thus usually unrealistic in such a setup. However, if good simulation models are available, simulation can be used for generation of data for both training the learning system and the performance of evaluation. In the following section, we describe the generation of the database.

A. Database

The tactile database aims to include numerous stable and unstable grasps on different objects. Ideally, the database allows us to investigate the two aspects of grasp stability recognition: shape specific and general shape stability recognition.

Shape specific recognition assumes that the system has approximate knowledge about the shape and/or pose of the object such that the grasps are in general reasonable, although they might still fail because of the uncertainties, as discussed in the introduction. This approach has practical real-world applications especially in cases where the manipulated object is located by a vision system. The data used for learning stability in shape specific recognition should concentrate around reasonable grasps generated by traditional grasp planning.

In general stability recognition, no knowledge of the object except its approximate position is used by the recognition system. Thus, the stability will be determined solely from the tactile input and the hand configuration. The data used for learning should contain examples of all possible situations and such a spherical sampling of grasp hypotheses would be useful. For general recognition, a significant amount of data over a large variety of objects is necessary. Due to time and space constraints, this work is focused on shape specific recognition and the generalization capability is studied with only a few objects. This paper uses the 3-finger Schunk Dextrous Hand (SDH) with 3 tactile array sensors and 7 degrees of freedom to generate grasping data. The plan is to extend the database with other popular hands.

Fig. 1 shows examples of objects that are included in the database. For each object, 10000 grasp attempts (including successful and unsuccessful ones) are stored. The two objects used in the experimental evaluation in this paper are objects c and d which are both cylindrical shaped. We expect to add more data of both simple and complex objects in the near future.

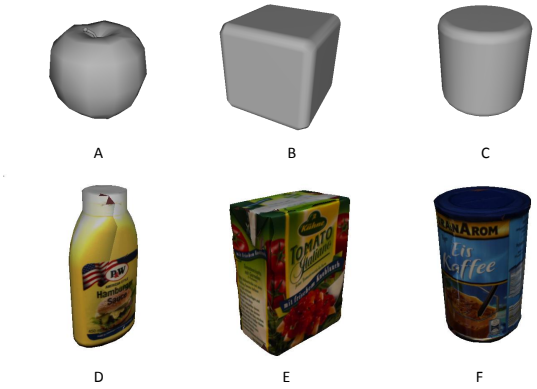


Fig. 1. Objects used in simulation

B. Simulation

The simulator described in [12] is used. The simulator can be used in combination with the Open Dynamics Engine (ODE) physics engine and provides support for simulating articulated hands, synchronous and asynchronous PD velocity ramp controllers, different grasp quality measures, camera sensors, range sensors and tactile sensors. The primary motivation for using our simulator over the more widely used GraspIt!, was the integrated support for tactile array sensors.

1) Tactile sensor model: The sensor simulation relies on the user to provide a function that describes the deformation of the sensor surface given a point force working perpendicular to it. The model assumes that the deformation or response is linear with the magnitude of the point force, which is a fair assumption for small forces. Given the deformation function $f(x, y)$ where x and y are specified relative to the center (a, b) of the contact, the total deformation of the surface of an array of rectangular texels with size (A, B) can be found by integrating over the surface of each texel by

$$g_{m,n}(a,b) = \int_{(A-\frac{1}{2})m-a}^{(A+\frac{1}{2})m-a} \int_{(B-\frac{1}{2})n-b}^{(B+\frac{1}{2})n-b} f(x,y)dxdy \quad (1)$$

where (a, b) is the center point of the contact and (m, n) is the texel index. This surface integration is approximated using the rectangle method. Point force experiments on the real sensors suggested that the deformation decreased with the inverse of

the square of the distance from the point force. We use an isotropic function to approximate the deformation of the sensor surface

$$f(x,y) = \zeta + \beta x + \frac{\alpha}{1 + x^2 + y^2} \quad (2)$$

where (x, y) is specified relative to (a, b). The parameters (α, β, ζ) were found by fitting the model to experimental data extracted from real sensors.

Fig. 2 shows a visual comparison between the real and the simulated sensor output where a sharp edge was pressed against both sensors.

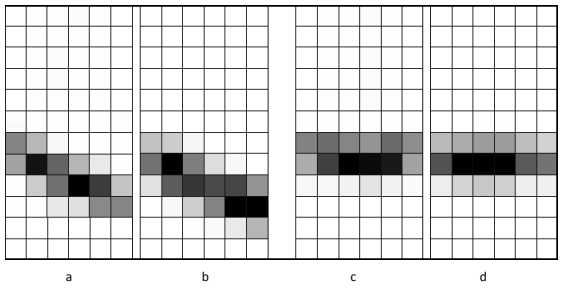


Fig. 2. Measured (a and c) versus simulated (b and d) sensor values. The tactile images were generated by pressing a sharp edge onto the sensor surface.

2) Grasp planning/selection: As described in Section III-A, the database includes data for several grasping strategies. For the general recognition, the approach directions for the hands were sampled on the unit sphere with origin in the object center of gravity.

For the shape specific recognition, the grasping strategies vary for each shape. The hand preshapes were generated with finger joint values in the interval $([-90; -70], [-10; 10])$ and where the 7'th joint was one of $90°, 60°, 0°$ as shown in Figure 3.

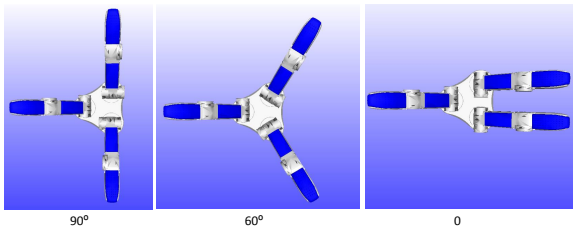


Fig. 3. Hand configuration of the 3-finger when the 7'th joint is at $90°$, $60°$ and $0°$

The grasp strategy for each shape is as follows:

- sphere - The approach directions are sampled randomly from the unit sphere with origin in the center of gravity of the object. The preshape is a ball grasp where joint 7 is $60°$.
- cylinder - The object is approached either from the top or from the side. When approaching from the top, a ball grasp preshape is used where joint 7 is at 60 degrees and the approach direction is pointing towards the center of gravity. For side grasps, the approach is sampled with an angle of 0-20 degrees with respect to the horizontal plane, pointing towards the center of mass of the object. The preshape in the side grasp uses an angle of 0 on joint 7, so that a parallel grasp is obtained.
- box shape - The object is approached using a vector lying in the plane defined by the world z-axis and the longest axis of the box and pointing toward the center of gravity. A parallel type preshape of the hand is used.

3) Grasp stability: In a robotic system, the stability of a grasp depends on several factors including

- Hand - orientation, joint configuration, friction, elasticity and grasping force.
- Object - shape, mass, friction, contact locations and area, and contact force.

In the simulated environment these parameters are known and we are therefore able to calculate the quality of a specific grasp. We use a widely known grasp quality measure based on the radius, ϵ, of the largest enclosing ball in the unit grasp wrench space (GWS). We construct the unit GWS as proposed in [13] by calculating the convex hull over the set of unit contact wrenches $\mathbf{w}_{i,j} = [\mathbf{f}_{i,j}^T \ \lambda(\mathbf{d}_i \times \mathbf{f}_{i,j})^T]^T$, where $\mathbf{f}_{i,j}$ belongs to a representative set of forces on the extrema of the friction cone of contact i, $\mathbf{d}_i$ is the vector from the torque origin to contact i and λ weighs the torque quality relative to the force quality.

Force and torque are dimensionally different and it is not obvious how to determine λ. We therefore calculate force space and torque space independently and use the radius of the largest enclosing ball in each of these to give a 2 dimensional quality value $(\epsilon_f, \epsilon_\tau)$ for each grasp. Stable grasps are defined as those for which both quality values are within a threshold which has been set experimentally.

IV. One-shot recognition

In this section, we examine the learning of grasp stability based on a single haptic measurement. We begin by introducing the notation used through the paper.

- $D = [o_i], i = 1...N$ denotes a data set with N observation sequences.
- $o_i = [x_t^i], t = 1...T_i$ is an observation sequence.
- $x_t^i = [M_{f_k}^{i^t} j_r^{i^t}], k = 1, 2, 3, r = 1...7$ is the observation at time instant t in the i-th sequence.
- $M_{f_k}^{i^t} = m_{p,q}^{H_{f_k}^{i^t}}$ are the moment features extracted from the tactile readings on finger f_k at time instant t in the i-th sequence. Details are given later in this section.
- $j_r^{i^t}$ is a joint value at time instant t in the i-th sequence.
- $H_{f_k}^{i^t}$ are the tactile readings collected from finger f_k at time instant t in the i-th sequence.

A method for learning grasp stability based on only one haptic measurement, x^{t_n}, discards almost all of the data available during the whole grasping sequence, $x^{t_1}, \ldots, x^{t_n}$. However, if successful separation between unstable and stable grasps can be learned from multiple examples, which we show

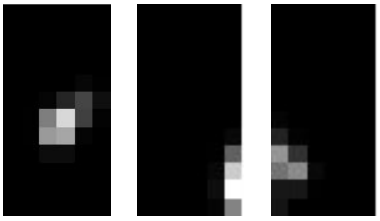
is possible within certain limitations, one-shot classification can determine the stability of the grasp from any haptic measurement x. The information gained from this one-shot classification can then be used for example in grasp control to determine when the robot hand has reached a stable configuration.

Next in Sec. IV-A, we describe the features used as an input for the support vector machine classification approach, which is then described in Section IV-B. Experimental evaluation is reported in Sec. IV-C.

A. Feature representation

The acquired raw haptic data, consisting of tactile readings $H^i_{f_k}$ and joint positions j_r, is high dimensional, $x \in \mathbb{R}^{223}$. The haptic data comprises of three tactile sensor readings, each giving 12x6 readings and 7 joint readings from the SDH.

Example images from the sensors are shown in Figure 4. The tactile images in the figure represent a stable grasp of a cylinder.



(a) (b) (c)

Fig. 4. Examples of tactile images from the three tactile sensors of the SDH: (a) First finger; (b) Second finger; (c) Third finger.

Because of the high dimensionality of the data, it is desirable to reduce the dimensionality by extracting meaningful features. While many feature extraction methods exist, taking into account the original representation of the tactile data, feature extraction techniques for images are strong candidates. For this reason, the use of image moments is proposed as a method to reduce the dimensionality of tactile images. Image moments are defined as

$$m_{p,q} = \sum_x \sum_y x^p y^q f(x,y) \ . \tag{3}$$

We compute moments up to order two, that is $(p+q) = o,\ o = \{0,1,2\}$. These are related to the total pressure, the location of the contact, and the shape of the contact area. Instead of the unnormalized moments, we normalize the zeroth order moment by calculating the average pressure $m_{0,0}/area$. First and second order moments are included in the feature vector as such. In addition, two extra features are computed for each tactile sensor/finger: The size of the contact area ($area$) and the center of contact $(\frac{m_{1,0}}{m_{0,0}}, \frac{m_{0,1}}{m_{0,0}})$. Thus in total there are nine features for each sensor, which are described as a single feature vector $\theta_t \in \mathbb{R}^{9s}$ where s is the number of sensors, $s = 3$ in the case of the SDH.

Normalization of the features usually improves the performance of machine learning approaches. For this reason, both moment based features and the finger joint angles are normalied to zero-mean and unit standard deviation. The normalization parameters (mean and standard deviation) are calculated from only the training data. The normalization is then applied equally for both the training and test data to normalize each invidual feature or dimension.

B. Support Vector Machine Classifier

As the problem of grasp stability is binary, support vector machine (SVM) classification [14, 15] is suitable for the problem. Thus, here the focus is on the 2-class SVM. SVM is a maximum margin classifier, i.e. the classifier fits the decision boundary so that maximum margin between the classes is achieved. This guarantees that the generalization ability between the classes is not lost during the training of the SVM classifier.

Another feature of the SVM is the ability to use non-linear classifiers instead of the original linear hyper-plane classifier. Non-linearity is achieved using different kernels, in this study radial basis function (RBF),

$$K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}, \quad for \quad \gamma > 0, \tag{4}$$

is used as the kernel for SVM. In addition to to the parameter γ, constant C, related to the penalty applied to incorrectly classified training samples [14], needs to be set. The parameters can be found by searching the parameter space to find the optimal values. In this study, as an extension to the basic two-class SVM, probabilistic outputs for SVM by Platt [16] are used to analyze the results given by the SVM.

C. One-Shot Recognition Results

The purpose of the experimental evaluation is to study the recognition performance of one-shot recognition in both a shape specific and a more general setting. In addition, the evaluation aims to increase the understanding of the recognition problem, that is, what seem to be the limitations of the one-shot recognition in general.

The data used for evaluation is from the database described in Section III. Three data sets, each describing a single object–grasp type combination, were used in evaluation. The used objects are a cylinder and a bottle, objects C and D in Fig. 1. The first data set involves side grasps on the cylinder, the second side grasps on the bottle and the third top grasps on the bottle. Small variations in the approach vector are used to vary the grasps, thus the data used is similar to noisy pose estimation. The number of samples is 6400, 4906 and 4446 for each data set respectively. Each set contains the same number of stable and unstable samples cases.

Results for both raw data (unprocessed tactile measurements) and the moment features described in the previous section are reported. In addition, we study both the case where the object and grasp type are known, and the case where more objects and grasp types have been used in learning, thus indicating the performance over a wider variety of objects and grasps.

The reported results are from 10-fold cross validation. The LibSVM [17] implementation of the SVM was used. The two parameters required by the SVM classifier, γ and C were set to 0.07 and 1 respectively. Raw features were normalized to $[0, 1]$ range as this normalization gave better classification performance for the raw features than the zero mean normalization.

1) Stability Classifier over Single Objects and Grasp Types: The classification results for individual data sets, i.e. per object or per grasp type classification, are shown in Table I, which shows the used features and data sets. The first row denotes the object type and the second row the grasp type.

TABLE I
SVM CORRECT CLASSIFICATION RATE FOR INDIVIDUAL DATA SETS.

	Cylinder Side Grasp	Bottle Side Grasp	Bottle Top Grasp
Raw	0.76	0.61	0.61
Moments	0.75	0.60	0.59

It is evident that the prediction of stability works relatively well for the cylinder but that the bottle is a more difficult object. The reason behind the differences in classification rates can be analyzed through the actual tactile images. The problem can clearly be seen from Figure 5. While the cylinder grasps show clear difference between the best and worst 5 percent of the grasps in terms of grasp quality measures, the bottle side grasps are overlapping. This overlap will only grow when considering more samples, thus, classification is difficult with this data. The same phenomenon affects the bottle top grasp as well.

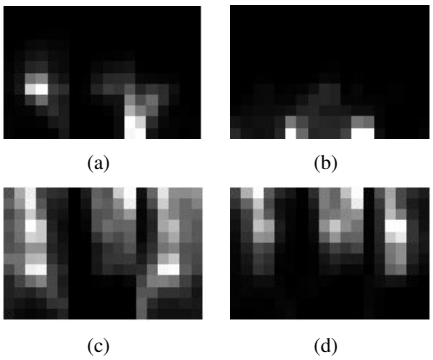


Fig. 5. Comparison of best and worst grasps of cylinder side grasps and bottle side grasps: (a) Best 5 % cyl. grasps; (b) Worst 5 % cyl. grasps; (c) Best 5 % bottle grasps; (d) Worst 5 % bottle grasps.

2) Common Stability Classifier: While the classifiers for a single object or grasp type are useful only for that particular object or grasp type, combining the information from several different objects or grasp types, forms a single common stability classifier, which can be used to predict the stability on a wider variety of objects and grasp types.

To test this common stability classifier, the three data sets used in the previous section, were combined into one data set, D_c. D_c consists of 9999 samples, 3333 from each of the three data sets. The ratio of unstable and stable samples in D_c is the same as in the original data sets.

Test showed that combining the three data sets from the single classifier case, the classification rate was 73 % correct classification when using the raw features and 74 % when using moments. Table II shows the individual classification rates for each of the three classes using the common classifier trained on all three data sets.

TABLE II
COMMON CLASSIFIER RATES PER DATA SET.

	Cylinder Side Grasp	Bottle Side Grasp	Bottle Top Grasp
Raw	0.79	0.71	0.70
Moments	0.79	0.71	0.70

The results indicate that combining the haptic data from multiple data sets, which include different objects and grasp types, increases the correct classification rate. This is remarkable as it indicates that adding more objects actually improves the classification rate rather than decreasing it, as would be reasonable to assume. Based on more experiments, it was found out that the improvement is only seen when the cylinder data set is included in the training data. This suggests that adding clear (easily classified) samples of unstable and stable grasps can improve classification with more difficult cases where the unstable and stable classes are overlapping in the feature space.

While the classification results are far from perfect, it should be noted that perfect discrimination between successful and unsuccessful grasps is not necessary because acceptance threshold can be set to a desired level of failure. Thus, the system might reject more stable grasps while having very few unstable grasps classified as stable ones. This would trigger re-grasping with some of the stable grasps, but such behavior would seem preferable in cases where it would avoid failures.

V. TEMPORAL RECOGNITION USING HMMS

This section studies the learning of grasp stability based on temporal information using Hidden Markov models (HMMs) [18]. The basic idea is to construct two HMMs, where the first models stable grasps and the second unstable ones. Recognition of an unknown grasp attempt as stable or unstable can then be performed by evaluating the likelihood of both models and choosing the one with the higher likelihood.

For the HMM, we use the classical notation $\lambda = (\pi, A, B)$ where π denotes the initial probability distribution, A is the transition probability matrix

$$A = a_{ij} = P(S_{t+1} = j | S_t = i), i = 1 \dots N, j = 1 \dots N \tag{5}$$

and B defines output (observation) probability distributions

$$b_j(x) = f_{X_t|S_t}(x|j) \tag{6}$$

where $X_t = x$ represents a feature-vector for any given state $S_t = j$. The structure of an HMM can be ergodic (fully connected) or left-to-right, which will affect the structure of matrix A. In the following work, we evaluate both of these common models. For more details about HMMs and the HMM estimation method, we ask the reader to consult a standard text.

A. Modeling Observations

The estimation of the HMM model parameters is based on the classical Baum-Welch procedure. The output probability distributions are modeled using Gaussian Mixture Models (GMMs):

$$f_X(x) = \sum_{m=1}^{M} w_m \frac{1}{2\pi^{K/2}\sqrt{|C_m|}} e^{-\frac{1}{2}(x-\mu_m)^T C_m^{-1}(x-\mu_m)} \quad (7)$$

where $\sum_{m=1}^{M} w_m = 1$, μ_m is the mean vector and C_m is the covariance matrix for the m-th mixture component. The unknown parameters $\theta = (w_m, \mu_m, C_m : m = 1...M)$ are estimated to fit the model to the sequences of training observations $o = (x_1, ...x_T)$.

Initial estimates of the observation densities in (Eq. 7) affect the point of convergence of the reestimation formulas. Depending on the structure of the HMM (ergodic vs left-to-right), we use a different initialization method for the parameters of the observation densities. The two initialization procedures are denoted $Init_1$ and $Init_2$:

- $Init_1$: For a ergodic HMM, observations are clustered using k-means. Here, k is equal to the number of states in the HMM and each cluster is modeled with a GMM using standard expectation maximization. Initial parameters for the GMMs are found in the standard fashion using the k-means algorithm.
- $Init_2$: For a left-to-right HMM, each observation sequence is divided temporally into equal length subsequences. Then, each GMM is estimated from the collection of corresponding subsequences. Thus, the GMMs (representing the states) represent the temporal evolution of the observations. Initial parameters for the GMM estimation are found identically to $Init_1$.

B. Experimental Results

The purpose of the experimental evaluation on simulated data is firstly to evaluate if the use of temporal information improves the recognition compared to the case of observing only a single time instant. Secondly, different common HMM types and models are evaluated in order to guarantee that the chosen model performs well. Similar to the one-shot recognition, the temporal recognition was evaluated on the three data sets described in Section IV-C. 80% of the samples were used for training and 20% for testing.

Before studying the HMM, the suitability of a GMM as a model for the data was evaluated by one shot recognition. We argue that if the GMM performs relatively well in one shot recognition compared to the SVM method presented earlier, GMMs are suitable for modeling the measurements at single time instants. Figure 6 shows the one shot recognition rates using GMMs with different number of mixture components. While being slightly inferior to SVM, it can be seen in the figure that the recognition rates are comparable such that we believe that the modeling of observations using GMMs is a valid choice.

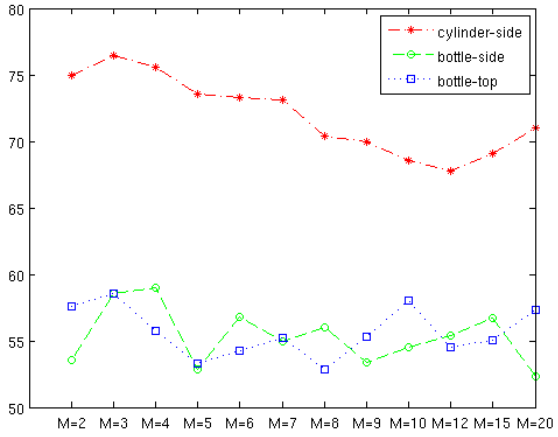


Fig. 6. One shot GMM recognition.

Next, to study if the temporal information improves the recognition performance, two HMMs, one for stable and another for unstable, were trained with the stopping criteria being the convergence threshold 10^{-4}. In order to improve the reliability of the evaluation, both ergodic and left-to-right HMM were evaluated independently with different structure parameters. The range of 2-6 for the number of states and 2-5 for the number of components in a mixture were evaluated. Finally, both spherical and diagonal covariance matrix structures were evaluated. The reason for these multiple experiments is that by evaluating multiple temporal models we aim to understand if the temporal sequence plays part in the understanding of the grasp stability, or if the final observation is sufficient.

Tables III and IV present the recognition rates for the ergodic and left-to-right HMMs. Ergodic and left-to-right HMMs have comparable results, while using the diagonal covariance matrix structure outperforms spherical distributions. The corresponding best parameter values are shown in Tables V and VI for the diagonal covariance matrix case. The recognition performace is comparable to one-shot classification indicating that the temporal aspect of the grasping action has little influence on the recognition performance. This result is somewhat surprising, it would seem probable that the additional information would be beneficial in the recognition. However, on the other hand the final measurements describe the final state of the system, which is also the state where the grasp stability is determined, which might explain the result.

To better understand the differences in performance for different objects, the distributions of logarithms of likelihood

TABLE III

BEST RECOGNITION RATES ON 3 DATA SETS, ERGODIC HMM

$Init_1$	Cylinder Side G.	Bottle Side G.	Bottle Top G.
diag cov	0.75	0.60	0.61
sph cov	0.74	0.60	0.60

TABLE IV

BEST RECOGNITION RATES ON 3 DATA SETS, LEFT-RIGHT HMM

$Init_2$	Cylinder Side G.	Bottle Side G.	Bottle Top G.
diag cov	0.75	0.60	0.61
sph cov	0.74	0.58	0.59

TABLE V

PARAMETERS FOR THE ERGODIC HMM AND DIAGONAL COV.

	Cylinder Side G.		Bottle Side G.		Bottle Top G.	
Name	St.	**Unst.**	St.	**Unst.**	St.	**Unst.**
State	5	4	6	5	5	6
Mixture Comp	4	4	4	3	4	3

TABLE VI

PARAMETERS FOR THE LEFT-RIGHT HMM AND DIAGONAL COV.

	Cylinder Side G.		Bottle Side G.		Bottle Top G.	
Name	St.	**Unst.**	St.	**Unst.**	St.	**Unst.**
State	6	4	2	4	5	5
Mixture Comp	4	5	5	2	4	4

ratios are shown for two objects for a well performing HMM model, the ergodic HMM with diagonal covariance matrices. Figure 7 shows the distributions for the cylinder side grasps, for which the performance was relatively good, while in Fig. 8 the distributions are given for the bottle side grasps, for which the stability was more difficult to recognize. Blue bars show the difference for stable samples and red bars are for unstable samples. It is evident in the figures that the stable and unstable grasps differ reasonably for the cylinder while the significant overlap of the distributions for the bottle indicates the fact that similarly to one shot classification, there are no clear differences in the tactile images during any part of grasping (compare to Fig. 5 which shows only the final time instant).

C. Experimental Results on Real Data

The main purpose of the real world experiments is to demonstrate that the grasp stability recognition is possible in real robots. In addition, we demonstrate that the same methods applied in simulation experiments also apply to the real world scenario. Thus, the experiments aim to serve as a proof-of-concept rather than assessing the exact performance rates in different use cases, which would require extensive data collection from each use case. Nevertheless, we believe that showing real world experimental results is important in order to validate the basic idea.

The grasping strategy for the experiments follows the methodology used in simulation such that the same objects and grasp types are used. The objects are placed such that they are initially not well centred with respect to the hand to investigate the capability of the learning system to cope with potential uncertainities in the objects' pose. A few example grasps are shown in Fig. 9. The hand prehapes shown in Fig. 3 were used with the 7th joint being 0° for side grasps and 45° for top grasps. After preshaping, the hand closes the fingers with the equal speeds and forces until reaching a static state where the object does not move or fully closed hand configuration is reached. The latter can occur only in the case of an unstable grasp.

Tactile readings and corresponding joint configurations were recorded starting from first contact until a static state is achieved. To generate the stable/unstable label for the grasp, the object is then lifted and rotated $[-120^\circ, +120^\circ]$ around the approach direction. The grasps where the object dropped or moved in the hand were labelled as unstable. The contacts between the object and the hand were on the tactile sensors on distal phalanxes. Data processing, training and classification followed the exact same methodology as used with the simulated data.

The total number of samples for the cylinder is 140, for bottle side grasps 100 and for bottle top grasps 50. The data was divided into separate training and test sets such that the number of unstable and stable samples are the same in the test

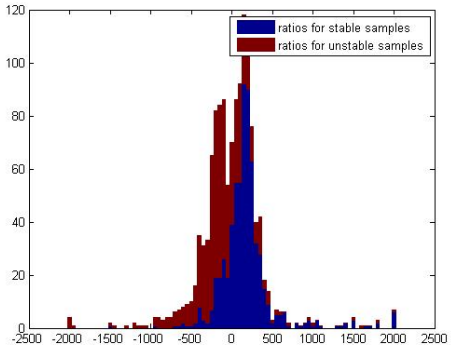


Fig. 7. The distribution of log-likelihood ratios for Cylinder side grasps.

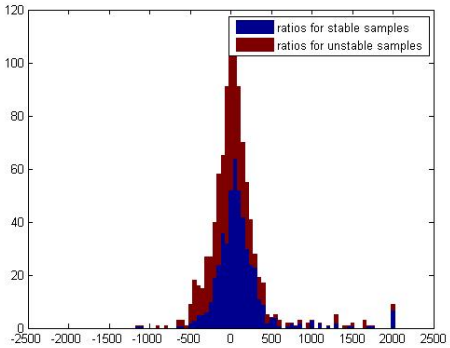


Fig. 8. The distribution of log-likelihood ratios for Bottle side grasps.

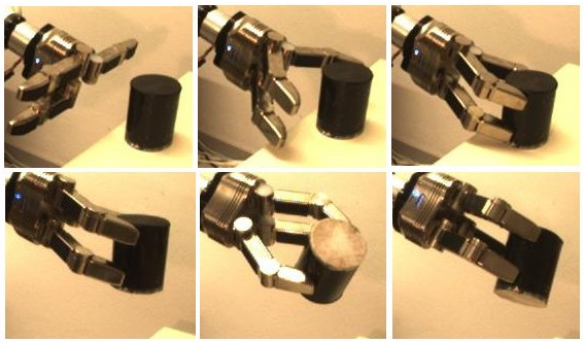


Fig. 9. A few examples from the execution of real experiments.

and train sets. The classification rates are shown in Table VII. First, it is evident that the idea of using the tactile feedback to evaluate the stability of a grasp is applicable also in a real world scenario. The recognition rates are higher compared to the simulated data. One reason is that the simulated data includes more variance in comparison to the real data. The goal of the experiments was however, to test the feasibility of the approach. More extensive experiments for the general shape stabilty evaluation is the objective of our current work.

TABLE VII
CLASSIFICATION RESULTS. RATES (STABLE (STATE, MIXTURE COMP.), UNSTABLE (STATE, MIXTURE COMP.))

	Cylinder Side G.	Bottle Side G.	Bottle Top G.
LR, diag	1 ((2,2),(3,3))	0.97 ((2,2),(2,2))	0.86 ((2,2),(2,2))
ER, diag	0.98((2,3),(5,4))	0.97 ((2,2),(2,3))	0.93 ((4,4),(3,3))
LR, spher	1 ((2,3),(3,2))	1 ((2,4),(6,4))	0.93 ((2,2),(2,2))
ER, spher	1 ((2,2),(3,4))	0.97 ((2,5),(2,3))	0.86 ((2,2),(2,2))

VI. CONCLUSION

This paper proposes the use of tactile sensing for estimating grasp stability. The experimental results show that even using a single tactile measurement allows relatively good recognition of grasp stability, and that the ideas studied in simulation are also applicable in real robot system. We do not aim for perfect discrimination between successful and unsuccessful grasps - this is not necessary because the acceptance threshold can be set to a desired level of failure. Thus, for example, the system may reject some stable grasps while having fewer unstable grasps classified as stable ones.

An important lesson to be learned from the experiments presented in the paper is that recognition rates improve when introducing clearly separable or "easy" cases of stable and unstable grasps. This applies for both the cases where the whole data set is clearly separable and those where the clearly separable samples are combined with more difficult samples of unstable and stable grasps. Thus, the wide variety of training data, not just the number of samples, seem to be important for good performance. Furthermore, the finding shows that the grasp stability can be generalized over objects to some degree, that is, introducing training data for some objects can improve the stability detection for other objects.

A somewhat surprising finding is that the use of time series data did not improve the results significantly. We believe that this is partly due to using relatively simple grasps in the database, where the objects are not moving significantly during the grasping attempt and the contacts do not change markedly. It is likely that in a more dynamic setting the time series approach would outperform the one shot recognition. However, this is out of the scope of the paper and remains an avenue for future research. We do not claim that the recognition approaches presented in the paper are the optimal ones. Our data will be made available and we invite other researchers to improve our results.

REFERENCES

[1] C. Goldfeder, M. Ciocarlie, and H. D. P. K. Allen, "The Columbia Grasp Database," in *IEEE International Conference on Robotics and Automation*, 2009.

[2] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Springer, 2008.

[3] A. Morales, M. Prats, P. Sanz, and A. P. Pobil, "An experiment in the use of manipulation primitives and tactile perception for reactive grasping," in *Robotics: Science and Systems (RSS 2007) Workshop on Robot Manipulation: Sensing and Adapting to the Real World*, Atlanta, USA, July 2007.

[4] M. Prats, P. Sanz, and A. del Pobil, "Vision-tactile-force integration and robot physical interaction," in *IEEE International Conference on Robotics and Automation*, Kobe, Japan, 2009, pp. 3975—3980.

[5] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic Grasp Planning Using Shape Primitives," in *IEEE International Conference on Robotics and Automation*, 2003, pp. 1824–1829.

[6] C. Goldfeder, P. K. Allen, C. Lackner, and R. Pelossof, "Grasp Planning Via Decomposition Trees," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 4679–4684.

[7] S. Ekvall and D. Kragic, "Learning and Evaluation of the Approach Vector for Automatic Grasp Generation and Planning," in *IEEE Int. Conf. on Robotics and Automation*, 2007, pp. 4715–4720.

[8] A. Petrovskaya, O. Khatib, S. Thrun, and A. Y. Ng, "Bayesian estimation for autonomous object manipulation based on tactile sensors," in *ICRA*, 2006, pp. 707–714.

[9] A. Jimneza, A. Soembagijob, D. Reynaertsb, H. V. Brusselb, R. Ceresa, and J. Ponsa., "Featureless classification of tactile contacts in a gripper using neural networks," *Sensors and Actuators A: Physical*, vol. 62, no. 1-3, pp. 488–491, 1997.

[10] M. Schöpfer, M. Pardowitz, and H. J. Ritter, "Using entropy for dimension reduction of tactile data," in *14th International Conference on Advanced Robotics*, ser. Proceedings of the ICAR 2009, IEEE. Munich, Germany: IEEE, 22/06/2009 2009.

[11] A. Schneider, J. Sturm, C. Stachniss, M. Reisert, H. Burkhardt, and W. Burgard, "Object identification with tactile sensors using bag-of-features," in *In Proc. of the International Conference on Intelligent Robot Systems (IROS)*, 2009.

[12] Empty, "Empty."

[13] C. Ferrari and J. Canny, "Planning optimal grasps," in *IEEE Int. Conf, on Robotics and Automation*, 1992, pp. 2290–2295.

[14] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[15] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[16] J. C. Platt and J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*. MIT Press, 1999, pp. 61–74.

[17] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[18] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in *Proceedings of the IEEE*, 1989, pp. 257–286.

Evaluation of Feature Representation and Machine Learning Methods in Grasp Stability Learning

Janne Laaksonen, Ville Kyrki and Danica Kragic

Abstract— **This paper addresses the problem of sensor-based grasping under uncertainty, specifically, the on-line estimation of grasp stability. We show that machine learning approaches can to some extent detect grasp stability from haptic data only. Using data from both simulations and two real robotic hands, the paper compares different feature representations and machine learning methods to evaluate their performance in determining the grasp stability. A boosting classifier was found to perform the best of the methods tested.**

I. INTRODUCTION

Grasping a known object in a known environment with a known robotic hand is a tractable problem. But immediately, when some of the facts are unknown, the problem becomes much more difficult to solve. The problem studied here is how to estimate grasp stability when only haptic information is available. Thus, there is no explicit object model, but the system is learning from haptic images of stable and unstable grasps. We show that it is possible to some extent to recognize when a grasp is stable when given only the haptic information.

A number of different sensor modalities can be used to deal with the uncertainty from having an unknown object during grasp. With sensors, we can determine when the object is in contact with the hand, giving additional information besides the kinematic configuration of the hand. Tactile sensors are useful here, as they measure the force or pressure inflicted on the sensor matrix, giving the area of the contact as well as the total force.

To determine the grasp stability, the stability criteria must be linked to the haptic data. This can be done either analytically or through learning. In this paper, we study the use of learning for grasp stability evaluation where a system learns the measure of stability based on a number of examples. Through an experimental study, our aim is to assess the suitability of different feature representations and machine learning methods in the problem of learning grasp stability from haptic input. The focus of the study is to evaluate the grasp stability from a single haptic data instance using both discriminitive and generative classifiers and different feature representations from data-driven dimensionality reduction techniques to application specific feature extraction methods. The approach taken in this paper gives the benefit of detecting whether the grasp is stable or unstable at any instant during grasping. Both simulated and real data is used to determine the differences and similarities when comparing simulation with real platforms.

The paper is divided into six sections: Section II is a study of related work in the area of the paper, Section III introduces the different features for the classification, Section IV describes the machine learning algorithms used in the experiments and Section V contains the actual performed experiments. Finally Section VI concludes the paper with discussion and future work.

II. RELATED WORK

Grasp stability analysis by analytical means is a well established field. However, to analytically determine the grasp stability, the kinematic configuration of the hand and the geometry of the object must be perfectly known. Previous studies on this subject are numerous and [2] gives a detailed review. However, the references are useful only in cases where the environment is fully known. When this is the case, it is possible to determine if the grasp is either force or form closure grasp [3], which ensures the stability.

While there is currently little work directly comparable to our work, many have studied the use of tactile and other sensors in a grasping context. Felip and Morales [4] developed a robust grasp primitive, which tries to find a suitable grasp for an unknown object after a few initial grasp attempts. However, only finger force sensors were used in the study.

Apart from using tactile information as a feedback for low level control [5], tactile sensors can be used to detect or identify object properties. Jiméneza et al. [6] use the tactile sensor feedback to determine what kind of a surface the object has, which is then used to determine a suitable grasp for an object. Petrovskaya et al. [7] on the other hand use tactile information to reduce the uncertainty of the object pose, upon an initial contact with the object. In their work, a particle filter is used to estimate object's pose, but the tactile sensor used to detect contact with the object is not embedded in the gripper performing the grasping.

Object identification has been studied by Schneider et al.[8] and Schöpfer et al. [9] Schneider et al. show that it is possible to identify an object using tactile sensors on a parallel jaw gripper. The approach is very similar to object recognition from images and the object must be grasped

The research leading to these results has received funding from the European Community's Seventh Framework Programme under grant agreement n° 215821.

J. Laaksonen and V. Kyrki are with Department of Information Technology, Lappeenranta University of Technology, P.O. Box 20, 53851 Lappeenranta, Finland, jalaakso@lut.fi, kyrki@lut.fi; Danica Kragic is with the Centre for Autonomous Systems, Computational Vision and Active Perception Lab, CSC-KTH, 10044 Stockholm, Sweden, dani@kth.se

several times before accurate recognition rates are achieved. Schöpfer et al. use a tactile sensor pad instead of a gripper or a hand which could be used to grasp the object. [9] is a study on different temporal features which can be used to recognize objects.

To the best of our knowledge, no studies have yet been published on the use of tactile sensors to estimate the grasp stability. When the grasp stability is analyzed purely from the haptic data, it gives a great advantage compared to the traditional grasp analysis methods. The stability can be learned from examples providing a good ground to cope with the uncertainty in the process generally not studied in the case of analytic approaches.

III. FEATURE REPRESENTATIONS

A haptic data instance, $\mathbf{H} = [\mathbf{t}\ \mathbf{j}]$, consists of the tactile readings, $\mathbf{t}$, and of the grasp joint configuration, $\mathbf{j}$. Depending on the hand used, the dimensionality of both $\mathbf{t}$ and $\mathbf{j}$ changes. In this study, three different platforms are used:

- Simulated Schunk Dextrous Hand (SDH), 3 fingers each with 12x6 tactile elements , $\mathbf{t} \in \mathbb{R}^{216}$, $\mathbf{j} \in \mathbb{R}^7$
- Schunk Dextrous Hand, 3 fingers each with 13x6 tactile elements, $\mathbf{t} \in \mathbb{R}^{234}$, $\mathbf{j} \in \mathbb{R}^7$
- Parallel Jaw Gripper, PG70, 2 fingers each with 14x6 tactile elements, $\mathbf{t} \in \mathbb{R}^{168}$, $\mathbf{j} \in \mathbb{R}^1$

The dimensionality of $\mathbf{H}$ ranges from $\mathbb{R}^{169}$ to $\mathbb{R}^{241}$ with the listed platforms. The number of features in $\mathbf{H}$ can be considered large and potentially redundant. Thus, an effective method to reduce the dimensionality precedes the subsequent processing. Rest of the section describes the methods that are used to achieve this.

To provide an overview of the effect features have on the classification of the grasp stability, several types of feature representations are studied for training and classification. The features, denoted by $\mathbf{f}$, are derived from the tactile sensor data, $\mathbf{t}$. The features represent a variety of approaches from pure data-driven dimensionality reduction to application specific features. The features are computed from the tactile readings only while the joint configuration is used as is as a part of the haptic features.

A. Principal Component Analysis

Principal component analysis (PCA) is commonly used linear technique for dimensionality reduction. Here, PCA is computed using the covariance of the haptic data, $\mathbf{H}_{1,\ldots,n}$ and the resulting eigenvectors and eigenvalues,

$$C = cov(H_{1,\ldots,n}) \,, \tag{1}$$

$$V^{-1}CV = D \,. \tag{2}$$

Here, V represent the eigenvectors and D the corresponding eigenvalues. We chose the eigenvectors with the largest eigenvalues that combined explain 90% of the data. This results in ~ 60 eigenvectors.

B. Image Moments

Raw image moments are defined as

$$m_{p,q} = \sum_x \sum_y x^p y^q f(x,y) \,. \tag{3}$$

The moments are computed up to order two, that is $(p+q) = o,\ \ o = \{0,1,2\}$, These are related to the total pressure, the mean of the contact area, and the shape of the contact area, indicated by the variance in x- and y-axes. Moments are computed for all tactile sensors individually, thus $\mathbf{f} \in \mathbb{R}^{18}$.

Raw image moments are used in the experiments as normalized image moments did not produce better results. This observation might be due to the fact that, e.g. rotation invariant moments, are not useful for grasp stability learning, as each grasp is unique.

C. Histogram

Histogram representation on the tactile data represents binning of the force affecting each cell of the tactile matrix. This operation also removes all spatial information. Thus, the histogram only considers the distribution of the affecting force. Using 10 histogram bins, $\mathbf{f} \in \mathbb{R}^{10}$.

D. Spatial Partitioning

Spatial partitioning partitions the area of the sensor matrix and sums the affecting force in every cell of the sensor matrix in each of these partitions. In essence, this sub-samples the tactile image of each sensor matrix. Partitioning can be thought as opposite to the histogram operation, as partitioning retains the spatial information but loses some information of the force distribution. In the experiments, a 2x2 grid is used to partition the tactile image on each sensor, $\mathbf{f} \in \mathbb{R}^{12}$.

E. Local Binary Pattern

Local binary patterns (LBPs) [10] are used commonly for texture classification but also on face recognition. As its name suggests, local binary pattern codes local changes in a binary code. The local changes are found by thresholding the pixel neighbourhood by the value of the center pixel and checking which pixels are above the threshold. These binary codes are then added to a histogram, which is the final feature representing the original data. Images from all sensors are coalesced into one image and the LBP is applied to this image in the experiments. In the experiments, LBP produces a histogram where $\mathbf{f} \in \mathbb{R}^{59}$.

F. Row and Column Sums

Row and column sums is another form of spatial feature representation, where the colums and rows are summed independent of each other, thus, the resulting dimensionality of the feature representation is the sum of the tactile sensor dimensions, $i+j$, for each sensor,

$$sum_{c_i} = \sum_j t_{ij} \,, \tag{4}$$

$$sum_{r_j} = \sum_i t_{ij} \,, \tag{5}$$

where sum_{c_i} denotes the individual sensor columns and sum_{r_j} denotes the sensor rows.

IV. CLASSIFIERS

From a classification point of view, the problem of classifying grasp stability may be modelled as a classical two-class problem. Thus, the stability is classified as either stable or unstable. This is possible to implement with most of the basic classifiers without extending the theories behind them.

In the work presented here, a number of classifiers have been selected for the experiments. All the classifiers described represent different types of machine learning algorithms that help to understand the underlaying problem in grasp stability classification. In particular, we study both discriminative and generative approaches for classification.

A. Support Vector Machine

As the problem of grasp stability is binary, support vector machine (SVM) classification [11], [12] is suitable for the problem. Thus, here the focus is on the 2-class SVM. SVM is a maximum margin classifier, i.e. the classifier fits the decision boundary so that maximum margin between the classes is achieved. This guarantees that the generalization ability between the classes is not lost during the training of the SVM classifier.

Another feature of the SVM is the ability to use non-linear classifiers instead of the original linear hyper-plane classifier. Non-linearity is achieved using different kernels and in this study radial basis function (RBF) is used as the kernel for SVM:

$$K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}, \quad for \quad \gamma > 0, \tag{6}$$

In addition to the parameter γ, constant C, related to the penalty applied to incorrectly classified training samples [11], needs to be set. The parameters can be found by searching the parameter space to find the optimal values. In this study, as an extension to the basic two-class SVM, probabilistic outputs for SVM by Platt [13] are used to analyze the results given by the SVM. The implementation of the SVM is by Chang and Lin [14].

B. Gaussian Mixture Model Classifier

As the naive Bayes classifier assumes that the data is distributed according to some modelable distribution, it is not optimal in cases where this assumption is not true. The haptic data is distributed according to an unknown distribution, thus it is reasonable to use Gaussian mixture model (GMM) statistical classifier.

While GMM methods assume a Gaussian distribution, GMM uses multiple Gaussian distributions to model the data which enables the methods to model multi-modal and more complex data. The implementation used in the experiments is by Paalanen and Kämäräinen [15].

TABLE I
TABLE OF PARAMETERS FOR FEATURES.

Features	Parameter	Parameter
Raw	-	-
PCA	-	-
Histogram	No. bins: 10	-
LBP	Uniform LBP	Samples: 8,1
Moments	-	-
Partitioning	Grid: 2x2	-
R&C sums	-	-

C. k-Nearest Neighbour

k-nearest neighbour [16] classifier is a very simple algorithm to implement. This classifier requires no training phase, instead during the classification phase, the test samples are compared to all given training samples. The test sample is classified as the class with the most neighboring, i.e. closest, training samples. The k denotes the number neighbouring training samples that are used in the classification phase. *k*-nearest neighbour also has a proven [16] error rate that is no worse than two times the error rate of an optimal classifier when the amount of data approaches infinity.

D. AdaBoost

AdaBoost or adaptive boosting is a meta-algorithm for learning which was developed by Freund and Schapire [17]. Adaboost uses multiple weak classifiers, such as linear hyperplane classifiers, to classify the given training data. AdaBoost has a good generalization ability, however AdaBoost is not effective when outliers are present in the training data.

The AdaBoost-algorithm that is used in this study is based on a decision tree classifier with a variable branching factor. With a branching factor of 1, the tree classifier represents a linear hyperplane classifier. The implementation is by Vezhnevets [18].

V. EXPERIMENTS

The goal of the experiments is to study the effect of the presented features in conjunction with multiple different classifier methods. A number of different datasets with different assumptions are used in the experiments to determine what type of data is suitable for classification.

A. Experimental Setup

The parameters for features and classifiers are shown in tables I and II. The raw data from the tactile sensors is also used as features in addition to the features presented in Section III. The parameters were found by a parameter search across reasonable parameter space. Objects used in the grasping experiments are shown in Figure 1.

The following datasets have been chosen from simulated data, which were generated using simulated SDH hand model in a simulation environment described in [19]:

- D_1, a cylinder, grasps sampled from the side
- D_2, a bottle, grasps sampled from the side
- D_3, a bottle, grasps sampled from the top
- D_4, a cylinder, grasps sampled from a sphere

TABLE II

TABLE OF PARAMETERS FOR CLASSIFIERS.

Classifier	Parameter	Parameter
SVM	C: 0.4	γ: 0.03
GMM	max. clusters: 19	max. error: 0.016
KNN	k: 3	-
AdaBoost	Branch factor: 1	-

- D_5, a bottle, grasps sampled from a sphere

The datasets $D_{1,2,3}$ represent cases where we know the pose of the object with some accuracy, and can plan for a grasp. The datasets $D_{4,5}$ are simulating situation were the position of the object is known to some extent but the orientation is unknown, thus, the grasp is sampled from a sphere around the object. In the simulated data, the grasp stability computation is based on [20], but instead of one convex hull W, two convex hulls, W_f and W_τ are used to separate wrench space with respect to forces and torques, and additional constraints are placed on W_f, so that

$$\alpha(m \cdot \mathbf{g}) \in W_f \, , \alpha = 1.1 \, . \tag{7}$$

This allows the grasp to remain stable even if some additional forces are introduced in addition to the gravity.

Datasets generated with real hands are following:

- D_6, a cylinder, grasps sampled from the side, SDH
- D_7, a bottle, grasps sampled from the side, SDH
- D_8, a bottle, grasps sampled from the top, SDH
- D_9, a box, grasps sampled from the side, PG70
- D_{10}, a shampoo bottle, grasps sampled from the side, PG70
- D_{11}, a shampoo bottle, grasps sampled from the top, PG70

Datasets $D_{6,\cdots,11}$ represent cases where an estimate of the object's pose is known, for example, from a vision system. This estimate is commonly noisy and thus we added the noise to the hand pose. The objects in datasets $D_{6,7,8,9}$ are rigid and the objects in datasets D_{10} and D_{11} are non-rigid, i.e. the objects are deformable. The grasp stability in these datasets was determined by grasping an object, after which it was lifted and rotated. If the object moved independently of the hand, the grasp was unstable, otherwise it was stable.

The method used to evaluate the performance of the classifiers was 10-fold cross validation. The dataset sample size for each of the given datasets are shown in Table III with the maximum classification rate summarized from Tables IV and V. The sample size shown in the table is balanced, so that each dataset has equal amount of stable and unstable grasp samples. All other features were normalized to zero-mean and unit variance, except the raw features which were normalized to range $[0,1]$. The normalization parameters were obtained from the training set and applied to both training and test sets.

B. Experimental Results

Result matrix with the described datasets is given in Table IV and Table V. The table shows the classification rate of each dataset with the indicated feature and classifier combination. Each row shows the best classifier in **bold** font and worst in *italic* font. The best and worst classifiers were determined on a 95 % confidence interval using the Agresti-Coull interval which approximates the binomial confidence interval. Multiple classifiers are deemed best if there is no statistically significant difference in the classification performance between them. Some results for GMM are omitted because of the training sample size requirements, thus, results for datasets $D_{6,\ldots,11}$ are not shown.

TABLE III

DATASET SAMPLE SIZE.

Dataset	Sample size	Max. classification rate
D_1	6400	0.770
D_2	4906	0.614
D_3	4446	0.627
D_4	5302	0.804
D_5	8990	0.705
D_6	140	0.921
D_7	100	0.921
D_8	50	0.846
D_9	148	0.746
D_{10}	148	0.590
D_{11}	100	0.640

The results in Tables IV and V show that there is a distinctive performance difference between the datasets. Simulated datasets, D_1 and D_4 perform usually better than the other simulated datasets. This performance gap is caused, at least partially, by the hand configuration, which allows the object to touch other areas of the hand where there are no sensors. This removes some of the important information about the object to be used in determining the grasp stability. Thus, it is important to set up the grasp sequence in a way that allows the sensored part of the hand to grasp the object.

This procedure is evident in the dataset gathered from the real hands, especially sets $D_{6,7,8}$, where the classification performance is above 75 % in some cases. However, the object in the datasets were rigid, which is not the case in sets $D_{10,11}$. These sets show mostly poor performance, indicating that further samples must be used to learn the grasp stability.

The best overall classifier is AdaBoost, which performs the best out of the four classifiers, while SVM is close second. Worst classifier is GMM, partially due to the extensive amount of data needed to train GMM successfully with some of the chosen features. Low amount of data available in datasets $D_{6,\ldots,11}$ makes it difficult to determine within the 0.95 confidence interval the best classifier, but looking at the results, AdaBoost has the highest mean in these cases. SVM has some anomalies, these are suspected to be caused by the parameter and feature combinations, and could be fixed by adjusting the parameters of SVM.

C. Feature Study

To study the effect of the features on the classification rate, tests with a 3-nearest neighbour classifier were conducted on each dimension of all the feature representations described in Section III and also on the raw tactile data. The dataset

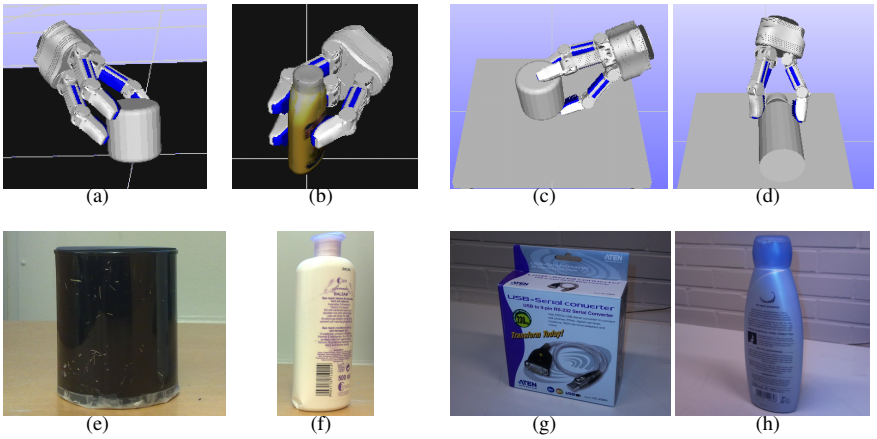


Fig. 1. Objects used in the datasets: (a) D_1; (b) D_2,D_3; (c) D_4; (d) D_5; (e) D_6; (f) D_7,D_8; (g) D_9; (h) D_{10}, D_{11}.

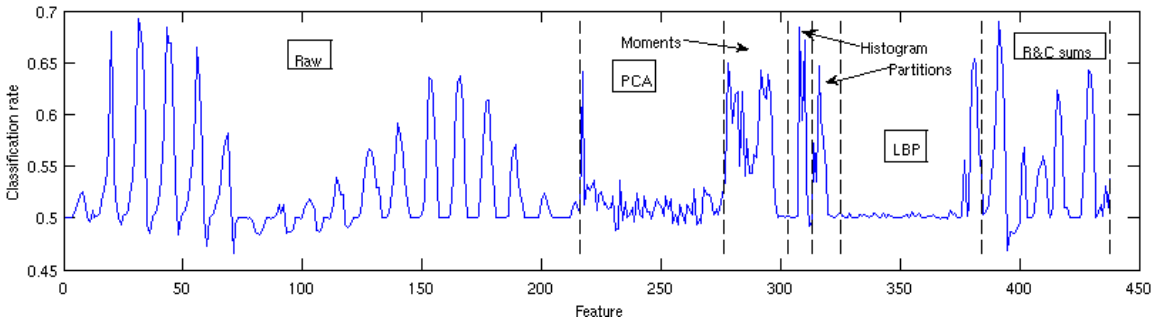


Fig. 2. Classification rates on individual features.

D_1 was used in this experiment. The classification rates are shown in Figure 2.

Classification rates of 0.5 or less in Figure 2 are a sign that the feature used is not particularly useful in learning as it has no correlation with the grasp stability. The figure shows that there are quite many useful features in the set of features that were tested. What is interesting is the raw data as it has multiple spikes which are among the best features for classifying the grasp stability. This indicates that individual cells of the tactile sensors can be used to determine the grasp stability to some extent. Also image moments, histogram and row and column sums seem to have a number of good features to use for classifying. However, it is important to note that these results are from a simulated dataset.

VI. CONCLUSIONS AND FUTURE WORK

The focus of the presented work was to investigate how different machine learning methods and feature representations affect the ability to learn and assess the grasp stability from haptic data. Both simulated and real world data was used in an experimental comparison. Experiments indicated that AdaBoost was the best performing classifier, suggesting that boosting approaches would be likely candidates for further studies in the context of grasp stability learning.

The classification performance varied significantly between different data sets. Results of the experiments showed that deformable objects are more difficult to learn with a similar sample size compared to rigid objects. A temporal approach might be useful for deformable objects, as it could extract more information from the grasp. Data also show that if the grasped object has contacts with the hand outside of the tactile matrices, the grasp stability can not be learned effectively. It needs to be noted that perfect classification performance is not necessary, since acceptance threshold can be set such that for example regrasping is triggered in ambiguous cases.

Future work will concentrate on expanding the presented study. Especially the study on deformable objects is interesting as currently there are no grasping simulators that are able to do this, but many household objects have this property. It is also possible to combine data from multiple objects to produce a common classifier for all the objects. Further research on this subject would help to identify the limits of the presented learning approach on completely unknown objects.

REFERENCES

[1] M. Ciocarlie, H. Dang, and P. K. Allen, "The Columbia grasp database," in *IEEE International Conference on Robotics and Automation*, 2009, pp. 1710–1716.

[2] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *ICRA*, 2000, pp. 707–714.

[3] D. Prattichizzo and J. C. Trinkle, "Grasping," in *Springer Handbook of Robotics*, 1st ed., B. Siciliano and O. Khatib, Eds. Berlin, Germany: Springer-Verlag, 2008.

TABLE IV

CLASSIFICATION RATES FOR DATASETS $D_{1,\cdots,11}$.

Data, Feature	SVM	GMM	KNN	AdaBoost
D_1	**0.755**	-	*0.733*	**0.767**
D_2	**0.591**	-	**0.566**	**0.581**
D_3	**0.603**	-	**0.607**	**0.621**
D_4	0.697	-	*0.631*	**0.793**
D_5, Raw	0.657	-	*0.582*	**0.689**
D_6	**0.826**	-	**0.907**	**0.914**
D_7	*0.220*	-	**0.840**	**0.910**
D_8	*0.493*	-	**0.846**	**0.804**
D_9	*0.540*	-	**0.713**	**0.711**
D_{10}	**0.493**	-	**0.486**	**0.464**
D_{11}	**0.500**	-	**0.540**	**0.560**
Mean	*0.580*		0.677	**0.710**
D_1	**0.770**	0.741	*0.725*	0.745
D_2	**0.597**	0.565	*0.561*	0.570
D_3	**0.613**	**0.604**	**0.597**	**0.604**
D_4	0.740	0.687	*0.675*	**0.776**
D_5, PCA	**0.676**	0.645	*0.604*	**0.677**
D_6	**0.857**	-	*0.586*	**0.900**
D_7	**0.770**	-	*0.550*	**0.690**
D_8	0.500	-	*0.479*	**0.786**
D_9	**0.732**	-	**0.655**	**0.717**
D_{10}	**0.500**	-	**0.540**	**0.540**
D_{11}	**0.460**	-	**0.550**	**0.490**
Mean	0.656	0.649	*0.593*	**0.681**
D_1	**0.765**	*0.711*	0.725	**0.759**
D_2	**0.611**	*0.527*	0.574	**0.586**
D_3	**0.627**	*0.540*	**0.601**	**0.623**
D_4	**0.800**	*0.612*	0.723	**0.797**
D_5, Moments	**0.705**	*0.518*	0.636	**0.689**
D_6	**0.921**	-	**0.936**	**0.907**
D_7	**0.910**	-	**0.860**	**0.920**
D_8	*0.271*	-	**0.671**	**0.779**
D_9	**0.646**	-	**0.695**	**0.727**
D_{10}	**0.440**	-	**0.505**	**0.447**
D_{11}	**0.480**	-	**0.510**	**0.640**
Mean	0.652	*0.582*	0.676	**0.716**
D_1	**0.731**	*0.649*	0.656	**0.739**
D_2	**0.560**	**0.550**	*0.522*	**0.564**
D_3	**0.620**	*0.499*	0.576	**0.621**
D_4	**0.790**	0.719	*0.698*	**0.794**
D_5, Histogram	**0.679**	**0.668**	*0.621*	**0.685**
D_6	**0.900**	-	**0.814**	**0.900**
D_7	**0.840**	-	**0.760**	**0.820**
D_8	**0.661**	-	**0.736**	**0.725**
D_9	**0.630**	-	**0.538**	**0.573**
D_{10}	**0.576**	-	**0.492**	**0.590**
D_{11}	*0.380*	-	**0.620**	**0.570**
Mean	0.669	*0.617*	0.639	**0.689**

TABLE V

CLASSIFICATION RATES FOR DATASETS $D_{1,\cdots,11}$.

Data, Feature	SVM	GMM	KNN	AdaBoost
D_1	**0.761**	*0.656*	0.718	**0.758**
D_2	**0.573**	**0.550**	**0.562**	**0.573**
D_3	**0.626**	*0.532*	0.590	**0.608**
D_4	**0.804**	*0.657*	0.730	**0.797**
D_5,Partitions	**0.690**	*0.609*	0.647	**0.689**
D_6	**0.913**	-	**0.914**	**0.921**
D_7	**0.910**	-	**0.890**	**0.890**
D_8	*0.368*	-	**0.818**	**0.675**
D_9	**0.630**	-	**0.606**	**0.644**
D_{10}	**0.408**	-	**0.455**	**0.488**
D_{11}	**0.560**	-	**0.480**	**0.640**
Mean	0.659	*0.601*	0.674	**0.699**
D_1	**0.750**	*0.644*	0.686	**0.749**
D_2	**0.548**	0.520	*0.514*	**0.564**
D_3	**0.609**	*0.580*	0.581	**0.613**
D_4	0.752	0.664	*0.640*	**0.797**
D_5, LBP	0.664	0.587	*0.578*	**0.684**
D_6	**0.843**	-	**0.793**	**0.850**
D_7	*0.260*	-	**0.680**	**0.740**
D_8	**0.471**	-	**0.682**	**0.600**
D_9	**0.611**	-	**0.692**	**0.734**
D_{10}	**0.502**	-	**0.458**	**0.483**
D_{11}	**0.500**	-	**0.490**	**0.510**
Mean	*0.592*	0.599	0.618	**0.666**
D_1	**0.768**	*0.638*	0.721	**0.765**
D_2	**0.614**	0.586	*0.578*	0.583
D_3	**0.627**	**0.615**	**0.615**	**0.607**
D_4	0.773	*0.589*	0.702	**0.796**
D_5, R&C Sums	**0.687**	0.634	*0.626*	**0.688**
D_6	**0.921**	-	**0.921**	**0.914**
D_7	**0.900**	-	**0.870**	**0.910**
D_8	*0.307*	-	**0.721**	**0.682**
D_9	**0.635**	-	**0.675**	**0.746**
D_{10}	**0.551**	-	**0.503**	**0.438**
D_{11}	**0.540**	-	**0.520**	**0.640**
Mean	0.666	*0.612*	0.678	**0.703**

[4] J. Felip and A. Morales, "Robust sensor-based grasp primitive for a three-finger robot hand," in *IEEE/RSJ International. Conference on Intelligent Robots and Systems*, Oct. 2009.

[5] T. Tsuboi and et al., "Adaptive grasping by multi fingered hand with tactile sensor based on robust force and position control," in *IEEE International Conference on Robotics and Automation*, 2008, pp. 264–271.

[6] A. Jiméneza, A. Soembagijob, D. Reynaertsb, H. V. Brusselb, R. Ceresa, and J. Ponsa, "Featureless classification of tactile contacts in a gripper using neural networks," *Sensors and Actuators A: Physical*, vol. 62, no. 1-3, pp. 488–491, 1997.

[7] A. Petrovskaya, O. Khatib, S. Thrun, and A. Y. Ng, "Bayesian estimation for autonomous object manipulation based on tactile sensors," in *ICRA*, 2006, pp. 707–714.

[8] A. Schneider, J. Sturm, C. Stachniss, M. Reisert, H. Burkhardt, and W. Burgard, "Object identification with tactile sensors using bag-of-features," in *In Proc. of the International Conference on Intelligent Robot Systems (IROS)*, 2009.

[9] M. Schöpfer, M. Pardowitz, and H. J. Ritter, "Using entropy for dimension reduction of tactile data," in *14th International Conference on Advanced Robotics*, ser. Proceedings of the ICAR 2009, IEEE. Munich, Germany: IEEE, 22/06/2009 2009.

[10] T. Ojala, M. Pietikinen, and T. Menp, "Multiresolution gray-scale and rotation invariant texture classification with Local Binary Patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[11] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[12] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[13] J. C. Platt and J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*. MIT Press, 1999, pp. 61–74.

[14] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[15] P. Paalanen and J.-K. Kämäräinen, *GMMBAYES - Bayesian Classifier and Gaussian Mixture Model ToolBox*, 2004, available at http://www2.it.lut.fi/project/gmmbayes/downloads/src/gmmbayestb/.

[16] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[17] Y. Freund and R. E. Shapire, "Experiments with a new boosting algorithm," in *Thirteenth Internation Conference on Machine Learning*. Morgan Kaufmann, 1996, pp. 148–156.

[18] A. Vezhnevets, *GML AdaBoost Matlab Toolbox*, 2006, aavailable at http://graphics.cs.msu.ru/ru/science/research/machinelearning/ adaboosttoolbox.

[19] J. A. Jorgensen and H. G. Petersen, "Usage of simulations to plan stable grasping of unknown objects with a 3-fingered schunk hand," in *IROS'08 Workshop on Robot Simulators*, Nice, France, September 2008. [Online]. Available: http://www.robot.uji.es/research/events/iros08/contributions/petersen.pdf

[20] A. T. Miller and P. K. Allen, "Examples of 3d grasp quality computations," in *IEEE International Conference on Robotics and Automation*, 1999, pp. 1240–1246.