Object Dexterous Manipulation in Hand Based on Finite State Machine

Qiang Li, Martin Meier, Robert Haschke, Helge Ritter and Bram Bolder

***Abstract*— We propose a simple but efficient control strategy to in-hand manipulate objects of unknown shape, weight, and friction properties. With this strategy, the object can be manipulated in hand in a large scale regardless there is rolling or sliding motion between the fingertips and object. We define several finger/fingers manipulation primitives and propose the hierarchical plan and control structure to facilitate the performing of the complex object manipulation task. The low level plan–local manipulation plan, is defined in continuous object configuration space, and fingers motion are planned in joints space according to the desired object motion and current perception feedback. The high level plan–global manipulation plan, is defined in finger gaits discrete space. We employ FSM (Finite State Machine) to modify fingers gaits to a new configuration in which new cycle low level plan will start again. In this way, we can solve the problem of robot hand workspace limitation. At last we design a four fingers manipulation in hand physics simulation experiment to prove the strategy feasibility. Simulation result shows the object manipulation result in ideal and simulated artificial noise cases.**

***Index Terms*— Multi-fingered Hand, Feedback-based Manipulation, Hierarchical Plan and Control**

I. INTRODUCTION

We consider the challenging task of dexterously manipulating an object within a multi-fingered robot hand, i.e. moving the object with respect to the hand.

There exists a considerable amount of work to analytically describe the motion of the object, finger tips and contact points during manipulation. These theoretical approaches assume various things to be known: the hand kinematics, object properties like shape, mass and mass distribution, the contact locations and friction coefficients, and the local surface geometry of both the object and finger tips. Based on this knowledge it is possible to compute joint-level finger trajectories in an offline fashion, and even determine slipping and rolling motions of the fingertips [10]. M.A.Roa[11] proposed to used ICR[6] idea to search feasible grasp points in continuous grasp space. With his method, object rolling and sliding manipulation can be planed. T.Phoka[9] extended the Roa's idea and searched feasible grasp points in hybrid grasp space, that is, not only continuous action(rolling and sliding) but also discrete action(finger gaits) can be planed. Their methods, however, are geometry-based plan and no robot hand kinematics was described, much work need to be done to extend their research to the real robot hand manipulation application. Because the friction cone is used to analyze the ICR, friction coefficient should be known in prior.

Comparing the analysis method, part recent approaches focused on the synthesis method. Researchers avoided the complex analysis of geometric relations and apply state-of-the-art motion planning methods like RRT [15] and PRM [12] to the manipulation problem. Xue et. al [14] tackled the problem of screwing a light bulb. Employing fast tactile feedback, Ishihara et. al [3] propose a control law to spin a pen of known shape at an impressive speed. Tahara et. al [13] point out a method to manipulate objects of unknown shape. They use a virtual object frame determined by the triangular finger-tip configuration of a three-fingered hand to derive a control law to manipulate the object's pose. However, without explicit sensory feedback, their method is limited in accuracy.

In order to solve the robot hand workspace limitation problem, finger gaits should be explicitly considered. New contact points should be selected in order to facilitate the new cycle local manipulation. phoka et. al [8] used heuristic approach to cluster the large number of discrete contact points to find the representive stable regrasp points. And this feasible regrasp points will be orgnized in the form of graph. In this way, it is easy to search and find a path from initial grasp state to the final grasp state. furukawa et. al [2] used three fingers robot hand, fast vision feedback and the predefined grasp posture to re-grasp the thrown object. They use vision to estimate the contact position of middle finger.

Comparing the previous *known* object manipulation in hand, we propose to employ the feedback based manipulation strategy to realize the *unknown* object manipulation in hand. We devide the object manipulation process into two stages: a local manipulation and a globally acting regrasp manipulation. The local controller reactively moves the object by a small amount only. Regrasp planning is employed to adapt the grasp configuration. Subsequently, local manipulation is continued. In feedback based manipulation strategy, what we need are vision feedback to estimate the object pose, joints angle and tactile feedback to estimate the contact position on the object. Currently, we obtain this feedback from a physical simulation, which is used to show the feasibility of the approach. To confirm the applicability of our method in noisy real-world feedback, we add artificial noise to the accurate sensor readings obtained from simulation.

Our previous work [5] has shown the feasibility of unknown object local manipulation feasibility. In this paper, we consider about using FSM to manage the finger gaits to realize the large scale manipulation. We also discuss

Q. Li, R. Haschke and H. Ritter are with the Research Institute for Cognition and Robotics (CoR-Lab), Bielefeld University, Germany {qli,rhaschke,helge}@cor-lab.uni-bielefeld.de

M. Meier is with the Neural Information Group, Bielefeld University, Germany mmeier@techfak.uni-bielefeld.de

B. Bolder is with the Honda Research Institute Europe (HRI-EU), Offenbach, Germany bram.bolder@honda-ri.de

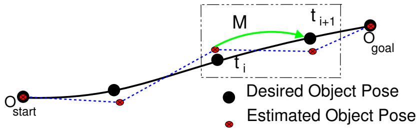


Fig. 1: Incremental manipulation of object pose O.

about the implementation of DMPL (Dexterous Manipulation Primitives Library) and introduce how to use the structural task command to formalize the finger/fingers primitives. Based on the primitives, complex tasks can be easily coded into programming. This software structure has two advantages: (1) to facilitate the hierarchical task planning [7], (2) to facilitate the complex manipulation tasks exploration learning [1].

The paper is arranged as following. In section II, we summarize the manipulation strategy and hierarchical plan and control structure. In section III, we introduce dexterous tasks which the multifingered robot hand can perform and DMPL. we also introduce how to code the finger/fingers primitives into programming, in which way it is easy to realize the new complex task implementation based on the exist finger/fingers primitives. In section IV, the physical simulation setup used for evaluation is introduced and simulation results are shown. Finally, section V summarizes our work.

II. Object Manipulation in Hand Strategy

Conventional grasp and manipulation planning methods uncoupled the planning from the control stage. The planning stage strongly depended on global knowledge about the geometry and friction issue of the object and fingertips.

In the strategy, we use the point contact model and do not explicitly model friction properties since the geometry and friction issue of object surface are unknown. However, in the simulation environment, the physical simulation engine adopts a Coloumb friction model approximating circular friction cones by four-sided pyramids to simulate the real physical contact friction. We employed micro manipulation assumption and rapid feedback loop to plan contact position and force, not the accurate contact interaction geometry plan.

We assume that we can estimate the current 3D pose O of the object, and object's target pose O' comes from high level global planner. eg. desired task, the decomposed subtask and instantiated action primitives. The pose is defined in the continuous Cartesian space and it can be described using arbitrary object Cartesian space configuration description method(homogeneous transform matrix, Euler angle or quaternion).Here we use homogeneous transform matrix description because it is easy to integrate this description into the exist matrix calculation library and also easy to transfer to the Euler angle, which will facilitate the visualizing the manipulation result. Based on O and O', we derive the required object motion M to realize the target pose within the next control cycle (cf. Fig. 1). Current local manipulation controller works in the *pure reactive* mode and it does not explicitly consider the reachable problem of the next

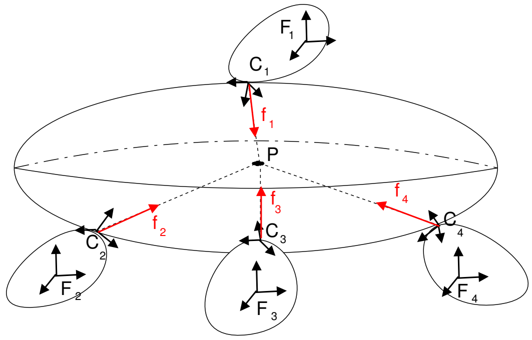


Fig. 2: Force planner employs centroid $\bar{\mathbf{p}}$ of contact locations.

desired object's pose in the continuous Cartesian space. This working mode can work well when the object surface is rotary, continuous and no edge occurring during the course of manipulation. While the complex object(eg.polyhedron) is manipulated, the robot hand should have cognition capability to predict the edge from the sensors feedback. Using this information to replan the manipulation task will be one part of our future work.

A. *Local Manipulation Controller–position*

The contact point position planing in Cartesian space is described as following: Denoting the current and targeted object pose with O and O' resp, we can easily compute the transformation matrix M describing the required finite object motion:

$$O' = O \cdot M \qquad \Leftrightarrow \qquad M = O^{-1} \cdot O' . \qquad (1)$$

Assuming, that contact positions do not move relative to the object within the control cycle, we can calculate the new contact positions $\mathbf{p}_i'$ (w.r.t. the palm). The contact positions $\mathbf{p}_i^o$ expressed relative to the object frame stay fixed:

$$\mathbf{p}_i' = O' \cdot \mathbf{p}_i^o = O \cdot M \cdot O^{-1} \mathbf{p}_i \, . \qquad (2)$$

From this we can compute the required positional changes $\Delta \mathbf{p}_i = \mathbf{p}_i' - \mathbf{p}_i$ for all contact points as input to the inverse hand kinematics. Because the local object and finger tip geometries as well as grasp stability measures are not explicitly taken into account, the actual grasp configuration might have changed after application of the computed hand pose. This corresponds to sliding or rolling contacts or even to a loss of a contact. On the contrary with the feedforward control which require one complete world model, our manipulation strategy just use the fast feedback to solve the unexpected rolling/sliding phenomenon.

B. *Local Manipulation Controller–force*

In order to maintain stable contacts anyway, we apply a force-control scheme additionally to the position-controlled object manipulation. Conventional contact force planners strive for a globally optimal contact force distribution ensuring grasp stability, i.e. all contact forces staying within corresponding friction cones, the totally applied force exactly resisting external forces (e.g. gravity), and limiting local contact forces. This general solution is meaningful only if the contact force is controllable. However, we assume that there is no 3D contact force feedback (obtained directly or indirectly), but only the force magnitude is available from tactile sensors. Following concepts from [13] the central idea is to plan the force direction such that the resultant moment will be zero, and to plan the force magnitudes along these directions such that the resultant force applied to the object becomes zero. Obviously the resultant moment is zero, if the contact force directions of all fingers intersect in one point. The force planner is illustrated in Fig. 2. The output of force planner is desired contact force along the *contact direction*, so the deviation force between current contact force and desired force can calculated. Two deviation from position planner and force planner are sent to composite position/force controller to calculate the composite contact position deviation which will be sent to Inverse Kinematics Module. One thing needed to be paid attention is that composite controller is composed of P/PI controller in position and force channel respectively in order to keep the no static error in contact force closed loop control.

All components of the local manipulation controller and robot hand angle angle position servo control are summarized in Fig. 3. More detailed description about the local manipulation controller can be found in [5].

C. Regrasp planner

Local manipulation controller can only modify the object pose in a small scale since the workspace of multifingered hand is limited. It's necessary to employ a global planner to change grasp posture to a new *"comfort"* configuration in which new fine manipulation can be possible. Our global regrasp planner was inspired by human manipulating object experience–using three passive fingers continuously rotating the object a small scale (eg.10 degree), one finger actively regrasping object in the appropriate point which can facilitate the new cycle passive fingers rotation. Global regrasp planner can be structured as Fig. 4. The global planner will work in discrete domain. It will not process the low level sensor feedback but the abstract representation of raw feedback. We use state-action space defined in the semantic level to describe the planner. In action space:

$A1$: **TFMR Rotate** $A2$: **TFM Rotate**
$A3$: **R Grip** $A4$: **R Exploration**
$A5$: **TFR Rotate** $A6$: **M Grip**
$A7$: **M Exploration** $A8$: **TMR Rotate**
$A9$: **F Grip** $A10$:**F Exploration**
$A11$: **FMR Rotate** $A12$: **T Grip**
$A13$: **T Exploration**

T: Thumb; F: Forefinger; M: Middlefinger; R:Ringfinger. "Exploration" means that the finger serves as exploration role and contact the object with very small contact force and explore the neighbor feasible contact points. "Grip" means that the exploration finger change it role to grip finger and contact the object with desired contact force planed by contact force planner. "Rotate" means that grip fingers

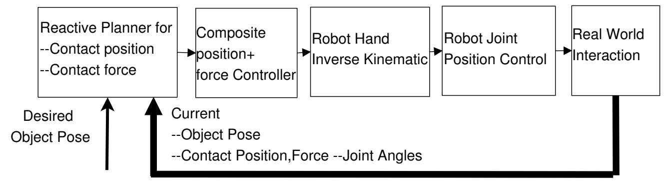


Fig. 3: Low level:local manipulation controller.

locally rotate the object.
In state space:

$S1$: **All are grip fingers**
$S2$: **TAM grip fingers and R exploration finger**
$S3$: **TFR grip fingers and M exploration finger**
$S4$: **TMR grip fingers and F exploration finger**
$S5$: **FMR grip fingers and T exploration finger**

The whole object manipulation process is shown in Fig. 4, which can be formalized as a FSM. This is a closed statechart and it's possible to transit from current state to the other arbitrary state in the statechart. Such state transition must pass by the medium state–four finger holding the object.

FSM starts from all fingers grasping the object in a *comfort* manipulation posture and all fingers are defined as grip fingers. Every finger use the same role switching process, so only ring finger switch is analyzed here as an example. Initially manipulation state is S1, and action 4 (A4) is taken after high level task is given(eg.rotate the object 10 degree around the X axis in the reference frame). After this action is finished, the process goes into state 2 (S2) and ring finger serves as exploration finger and contacts the object with a small enough desired force(e.g 0.1). TFM are grip fingers and their desired force/position are planned by the local force/position planner to stably grasp object. Small enough contact force of exploration finger is used in order to not damage the other three finger stable grasping(they assume the contact force of ring finger is 0). It also provides the contact information about the exploration finger, which is helpful for searching new feasible grasp point. At state 2 (S2), action 2 (A2) is taken and TFM–three grip fingers are used to rotate the object. This is a local rotation manipulation, we can use the local manipulation algorithm developed in [5]. The exploration finger still contact the object with small contact force and try to explore its neighbor feasible contact points in the object surface. After a small rotation is finished, action 3 (A3) is taken and ring finger releases its exploration finger role and changes to be the grip finger. Its desired contact force is changed from small enough to the desired contact force planed by local contact force planner(for four finger grasp). In this case, manipulation process reaches back to state 1 (S1). Other fingers use the same principle to realize the finger role switch.

III. DMPL AND STRUCTURAL TASK DESCRIPTION

A. The definition of DMPL

In order to facilitate the global planner implementation, It will be helpful to refine the abstract primitive sets from the task on hand. DMPL provides such an abstract primitives

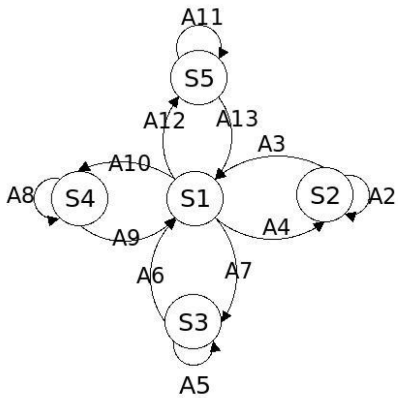


Fig. 4: High level:global regrasp planner.

set which can be defined in discrete domain and serve as the basic element for the high level dexterous manipulation planning. Currently these primitives include:

- Finger position control in Cartesian space.

This abstract primitive is used to drive the finger to approach one setting spatial point without the interaction with the object. It can be instantiated by assigning a spatial position parameter or a vision feature.

- Finger composite control in Cartesian space.

This abstract primitive is used to drive the finger to contact one setting spatial point with a setting contact force. It can be instantiated by assigning a spatial position and contact force parameter. This primitive is mainly used in the object manipulation.

- Finger hybrid control in Cartesian space.

This primitive is similar with composite control primitive, but using different control law to realize the instantiation primitive. Composite control primitive superimposed the output of force/position controller. Hybrid control primitive, however, uncouple the force and position planning in the contact frame. It plans contact point position in the contact tangent surface and plans contact force in the normal vector. This primitive is mainly used in finger exploring on the object surface.

Using the previous three primitive, object 3D small scale motion – one composite manipulation action(object-centered) can be realized. This action can be instantiated by assigning the desired object pose and setting the manipulation fingers.

B. Structural Task Description

DMPL have been coded in C++ successfully. Programmer can manually write action sequence based on such primitives to realize the complex manipulation task according to transition conditions(events) in structural environment. We show one human–robot hand interaction manipulation statechart which cover the tasks which can be realized by DMPL in Fig. 5. Firstly, we assume the object is fixed. This is a reasonable assumption because the object can be held by human and transfered to the robot hand in human robot interaction application. The robot hand can grasp the object by telling it predefined contact point on the object. Before the robot hand contact the object, it will use the finger position control primitive to drive the all manipulation fingers to approach the setting points. All fingers position control primitives are in the orthogonal states. Once contact is detected by the tactile sensor on the fingertip, the fingers composite control primitive is employed. When all fingers are in the composite control primitive and stable grasp condition are met, the manipulation will switch to the unfrozen object state. Human can release the object, and robot hand will hold the object in the setting configuration(predefined pose of the object). Then human can send command to robot hand to perform task (eg.small scale object manipulation, finger regrasp, and large scale object manipulation etc.). Such tasks are exclusive in current software version. At last, human can send command to end task.

This manually planner can work well in no unexpected event happen.It, however, can not robustly deal with the following *dynamic*, *open-end* manipulation environment.

- Irregular object surface and unknown manipulation task.

The principle of hard coding object manipulation is to transfer human manipulation intelligence to robot hand. Human model the manipulation process and translate this manipulation process into programming language. This method can work very well in the known object(human manipulated). It's difficult to generalize the manipulation process to the unknown object. Eg. If the robot hand has no idea about the object surface geometry, it will not know when it has to leave the one surface to stride across the edge of the object to contact a new surface. Which surface it should contact in order to keep the stable grasping. On the other hand, when the operation environment is open end, the robot hand does not know what's the next task. The programmer can not predict all tasks the robot hand need to performed and coding it in the compile time. When the new task is coming, the programmer has to re-write programming.

- Unexpectedly finger leave the object surface.

The phenomenon happen because of the conflicting between the kinematics planning domain and dynamics implementation domain. Sometimes although planning in kinematics level is correct, the execution in dynamics level is not so stable because of poor measure accuracy and model error.

In order to solve such problems, we propose to use the structural task description command and auto-replan mechanism. Auto-replan mechanism will replan task sequence while the contingence happen between the anticipation state and the current perception state, then the replan result will be automatically "translated" into the structural task description command. We have finished defining and coding such structural command, and how to model the manipulation behaviour, auto replan task and "translate" the task into the task description will be our future work. In order to formalize the structural task description, the main data structure is defined as in Fig. 6. Class FIstate is used to describe the finger state to show whether the finger contact the object, whether

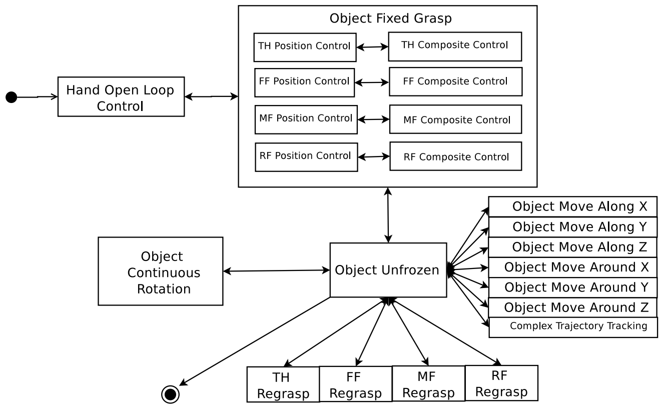


Fig. 5: Human Robot Hand Interaction Object Manipulation statechart

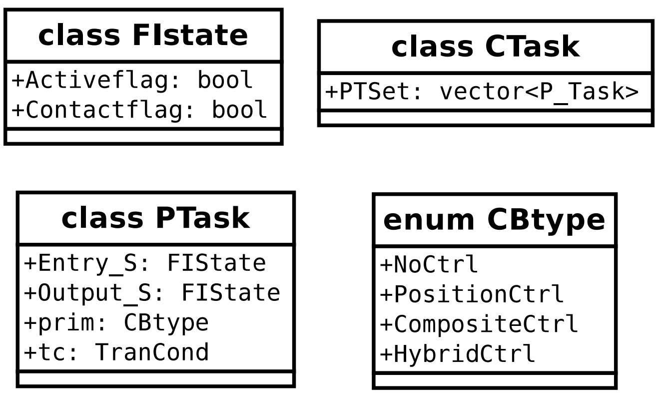


Fig. 6: Data structure definition

the finger is active finger or passive finger. Class PTask is primitive task definition. This class define:which primitive task is implemented, what's the entry state and output state while the primitive is implemented and what the transition condition of this primitive task is. CTask is complex task which are composed of the primitive task temporal sequence for every finger and for desired object motion. With this data structure, it is very convenient and intuitive to code the FSM described in Fig 4. Every subtask(action) can be structured as: desired entry state, desired output state, action primitive, transition condition(for finger), and desired object motion and how to interpolate this desired motion into tiny step(see in Fig. 1). The source code of DMPL and structural task description can be checked out in [4]

IV. SIMULATION

The object manipulation algorithm is validated in a physical simulation experiment. We use the Vortex physics engine to obtain real-time contact information (i.e. contact position and contact force magnitude), and the object's pose (object position and orientation). Artificial noise is superimposed on feedback provided by the physics engine. Currently two geometric primitives, namely sphere (radius=2.5cm), cylinder (radius=2.5cm, height=9cm) are evaluated. The objects are sized middle-scale compared to the robot hand, so rolling and slipping between the fingertips and the object will occur during the course of the manipulation. Object parameters information(radius of the ball and cylinder) is not available to the manipulation strategy. The simulation scenario is shown in Fig. 7 resembling our real robot setup to facilitate future transfer into real world, once the required tactile feedback is robustly available from finger tip sensors. The controller gains are manually set to guarantee the stability of the object manipulation.

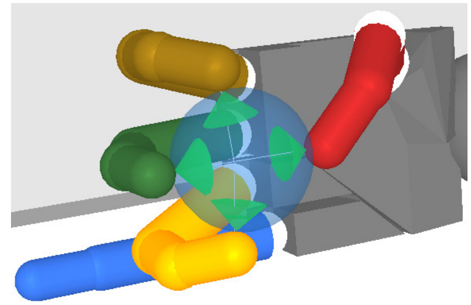
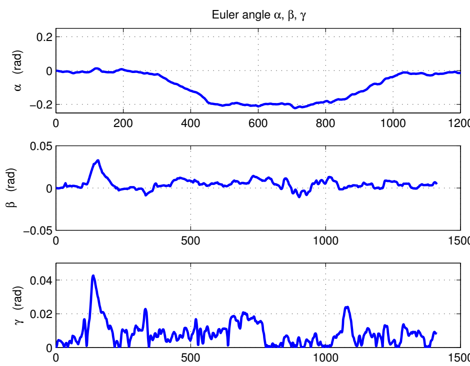
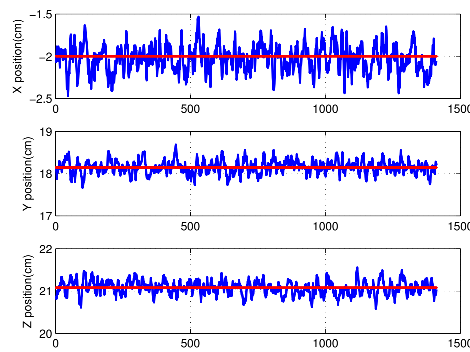


Fig. 7: Simulation scenario



(a) Orientation of the object



(b) Position of the object

Fig. 8: Object rotation around x-axis

We assume that the object has been successfully grasped and grasp points are comfort for the manipulation. Only the manipulation phase is considered in this simulation experiment.

In the following we present results for the exemplary manipulative motions to show the feasibility of our manipulation

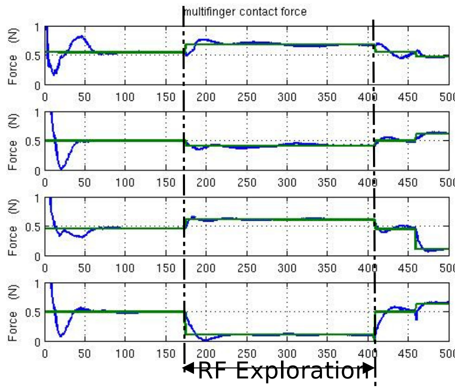


Fig. 9: Exploration(ring) finger contact the object.

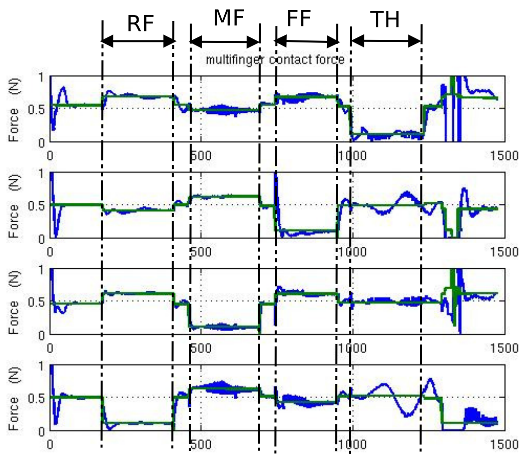


Fig. 10: Finger roles switching.

strategy.

1) Action primitive I - locally rotate the object with small rotational movement of 0.2 rad around the x-axis. The simulation result is shown in Fig. 8. In simulation, we superimposed artificial measurement noise to the object position value from the physics engine. The standard deviations of the added Gaussian noise are: 0.5cm. In Fig. 8a it showed that object is rotated from 0 to 0.2 rad and back again. In Fig. 8b blue solid line means the noisy object position feedback and red solid line means the nominal object position.
2) Action primitive II - exploration finger contacts the object with the desired force. In Fig. 9, ring finger is exploration finger and it contact the object with small contact force – 0.1. Other three fingers use the contact planner to design the desired contact force.
3) One periodical of finger role switching. The contact force result is shown in Fig. 10. Fig. 10 showed that ring finger, middle finger, forefinger and thumb served exploration finger in sequence to implement the continuous rotation of the object.

V. Summary

We propose a reactive control strategy to realize manipulation motions for unknown objects. In contrast to traditional manipulation strategies, which require a lot of information about the object and plan in an offline fashion, our plan method developed in local manipulation level and global manipulation level is in an online fashion and employs minimal sensory information. Abstract primitives are refined to facilitate the implementation of global manipulation planning. FSM is employed to manage manipulation state transition and improve the object manipulation robustness. In physical simulation experiments we proved the feasibility of the manipulation strategy.

VI. ACKNOWLEDGEMENT

Qiang Li gratefully acknowledges the financial support from Honda Research Institute Europe for the project "Autonomous Exploration of Manual Interaction Space".

References

[1] N. Furukawa, A. Namiki, S. Taku, and M. Ishikawa. Dynamic regrasping using a high-speed multifingered hand and a high-speed vision system. In *Robotics and Automation,IEEE International Conference on*, pages 181–187. Ieee, 2006.
[2] T. Ishihara, A. Namiki, M. Ishikawa, and M. Shimojo. Dynamic pen spinning using a high-speed multifingered hand with high-speed tactile sensor. In *Humanoid Robots,IEEE International Conference on*, pages 258 –263, 2006.
[3] Qiang Li. Autonomous exploration of manual interaction space. https://projects.cor-lab.de/projects/asemis.
[4] Qiang Li, Robert Haschke, Helge Ritter, and Bram Bolder. Simulation results for manipulation of unknown objects in hand. In *Robotics and Biomimetics, IEEE International Conference on*, 2011.
[5] Van-Duc Nguyen. Constructing force-closure grasps. *International Journal of Robotics Research*, 7:3–16, June 1988.
[6] Dominik Off and Jianwei Zhang. Continual htn planning and acting in open-ended domains - considering knowledge acquisition opportunities. In *Proceedings of the 4th International Conference on Agents and Artificial Intelligence*, pages 16–25, 2012.
[7] T. Phoka and A. Sudsang. Contact point clustering approach for 5-fingered regrasp planning. In *Intelligent Robots and Systems,IEEE/RSJ International Conference on*, pages 4174–4179. IEEE, 2009.
[8] T. Phoka and A. Sudsang. Regrasp planning of three-fingered hand for a polygonal object. In *Robotics and Automation,IEEE International Conference on*, pages 4328 –4333, may 2010.
[9] HANDLE Project. http://www.handle-project.eu/index.php/home/welcome.
[10] R.M.Murray, Z.X.Li, and S.S.Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.
[11] Mximo A. Roa, Ral Surez, and Jan Rosell. Grasp space generation using sampling and computation of independent regions. In *Intelligent Robots and Systems,IEEE/RSJ International Conference on*, pages 2258–2263, 2008.
[12] J.P. Saut, A. Sahbani, S. El-Khoury, and V. Perdereau. Dexterous manipulation planning using probabilistic roadmaps in continuous grasp subspaces. In *Intelligent Robots and Systems, IEEE International Conference on*, pages 2907–2912, 2007.
[13] K. Tahara, S. Arimoto, and M. Yoshida. Dynamic object manipulation using a virtual frame by a triple soft-fingered robotic hand. In *Robotics and Automation, IEEE International Conference on*, pages 4322 – 4327, 2010.
[14] Z. Xue, J.M. Zollner, and R. Dillmann. Dexterous manipulation planning of objects with surface of revolution. In *Intelligent Robots and Systems, IEEE International Conference on*, pages 2703–2708, 2008.
[15] M. Yashima. Manipulation planning for object re-orientation based on randomized techniques. In *Robotics and Automation, IEEE International Conference on*, volume 2, pages 1245–1251, 2004.