2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

# Description and Execution of Humanoid's Object Manipulation based on Object-environment-robot Contact States

Shunichi Nozawa[1], Masaki Murooka[1], Shintaro Noda[1], Kei Okada[1], Masayuki Inaba[1]

***Abstract*— In the case of object manipulation by a humanoid robot, it is important to deal with contact states between objects, a robot, and an environment both to avoid falling down and to achieve objective manipulations. We propose a method to describe and uniformly execute various object manipulations by a humanoid robot. In description, we focus on the contact states and define manipulation phases according to the contact states. In execution, the humanoid's controller autonomously switches manipulation phases and substantiates the contact-force controller. According to switching of the manipulation phases, the humanoid's manipulation system switches the inputs for the contact-force controller, which includes the estimation of object's information and motion generation. We evaluated our proposed system through experiments in which the HRP-2 robot manipulates four objects without information about the objects' masses and necessary operational forces.**

## I. INTRODUCTION

Recent research has achieved humanoid manipulation of objects by applying control theories suitable for each task; for example, lifting up [1], [2], pushing[3], [4], pivoting[5], and manipulating structured objects [6], [7]. These manipulations utilize the advantages of the physical structure of humanoid robots.

To develop a generalized system capable of the above-mentioned manipulations, the humanoid's system should adequately treat contact states between an object, a robot, and an environment which change during manipulation. The object information to achieve manipulation and maintain full-body balance is different for different contact states. For example, in the case of pushing an object, both the object and the environment support the humanoid robot so that the controller utilizes reaction forces from the object. On the other hand, in the case of lifting an object, the environment supports the robot so that the controller utilizes the object's mass properties as well as the humanoid's mass properties.

In this paper, we propose a uniform method to describe and execute object manipulations by a humanoid robot by focusing on the contact states. In our proposed method, a humanoid manipulation system uses a single motion controller by switching controller's parameters according to the actual contact states observation. We discuss a method to make the system support autonomous execution of manipulation, adaptivity to an unknown object's mass or unknown object's operational forces, and error recovery.

[1] S. Nozawa, M. Murooka, S. Noda, K. Okada and M. Inaba are with Department of Mechano-Infomatics, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan nozawa at jsk.t.u-tokyo.ac.jp

## II. DESCRIPTION AND EXECUTION OF HUMANOID'S OBJECT MANIPULATION BASED ON CONTACT STATES BETWEEN AN OBJECT, A ROBOT, AND AN ENVIRONMENT

### A. Problem

Contact states between an object, a robot, and an environment (ORE contact states) are different for the method of manipulation and the phase of manipulation. We can classify humanoid's manipulation into two parts:

($\alpha$) Contact-transition part
The humanoid's motion invokes the transition of contact states.

($\beta$) Contact-steady part
The humanoid's motion generates itself or object's motion at a contact state.

In this paper, we discuss a method to describe and execute both ($\alpha$) and ($\beta$). Especially, we focus on planning and execution of these, adaptation to unknown parameters of objects, and recovery from error.

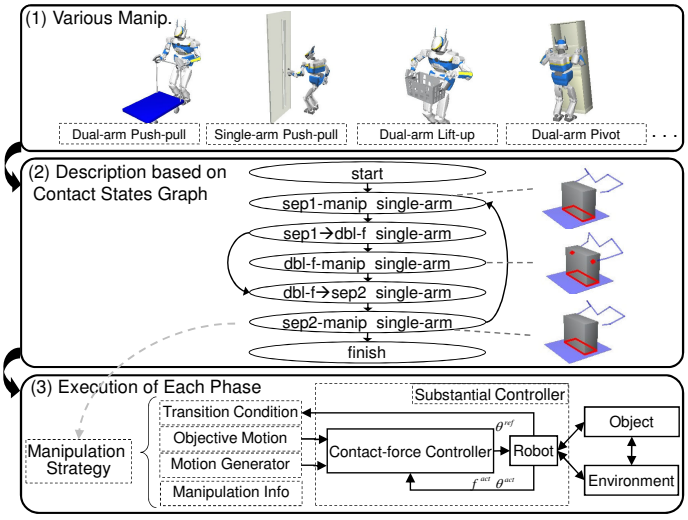



Fig. 1. Description and Execution of Object Manipulation

### B. Our Proposed System

We propose a method to describe and execute both ($\alpha$) and ($\beta$) in object manipulation by focusing on ORE contacts. Fig.1 shows our proposed system. We describe ($\alpha$) as a Contact States Graph (CSG) which represents contact states transition. We also describe ($\beta$) as object's or humanoid's motion at each node in the graph. The system inputs are symbolic information of manipulation for ($\alpha$) and objective motion of the objects for ($\beta$). We utilize planner for both ($\alpha$) and ($\beta$) instead of describing graphs and motions just

by manual. Graph generator plans a CSG (Fig.1 (2)) from symbolic information such as "dual-arm lift-up" (Fig.1 (1)). Motion generator plans humanoid's motion based on objective motion. The system executes manipulation by tracing nodes in a CSG for ($\alpha$) (Fig.1 (3)). According to the current nodes, the system switches necessary information for controllers, which we call Manipulation Strategy, and executes humanoid's motion using Contact-force Controller for ($\beta$).

In the following sections, we introduce these descriptions and executions in detail. In Sec. III, we introduce a method to describe manipulation using a CSG corresponding to Fig.1 (1) and (2). In Sec. IV, we introduce a method to describe motion at each node based on Manipulation Strategy (Fig.1 (3)). In Sec. V, we explain autonomous execution of manipulation by tracing a CSG (Fig.1 (3)), substantiation Contact-force Controller, and error recovery. In Sec. VI, we evaluate our proposed system through an experiment in which the HRP-2 robot manipulates four different objects.

### C. Related Works and Contributions of this Paper

In this subsection, we explain contributions of this paper compared with related works.

*1) Switching the behavior of humanoid's controller:* In this paper, we discuss both what a humanoid's system should switch during manipulation and what we can commonalize through different object manipulation. By using our proposed system, we can utilize the controllers discussed in related works [1] [4] [5] [6]. Methods to generate a humanoid's motion and to estimate necessary object information for each manipulation, correspond to the Manipulation Strategy in the system. In related works, Stilman et al. achieved adaptive pushing [4] and Harada et al. [1] discussed estimation of object's mass and center-of-mass (COM) in a humanoid's carrying-up motion. On the other hands, we can commonalize autonomous execution by switching Manipulation Strategies according to actual contact states observation and error recovery.

*2) Capability of describing both contact states and objective motion :* Our proposed method involves description of both ($\alpha$) and ($\beta$). Contact-graph-based description of manipulation have been discussed in the field of industrial manipulators and object assembly. We extend the description for a humanoid robot by considering Manipulation Strategies. Keith et al.[8] proposed a method to plan humanoid's motion in time-varing phases. We segment manipulation based on contact states to use adequate controllers' inputs. Kanehiro et al.[9] proposed a method to generate the humanoid's behavior by utilizing state transitions of sensor information such as joint angles. Although this method is useful for achieving adaptive behavior even if some errors occurs, it was difficult to describe quantitative objective motion such as locomotion. Here, we distinguishes ($\alpha$) and ($\beta$) to describe both various contact states transition and quantitative objective. Moreover, our system is applicable to error recovery both by switching contact states and by modifying Objective Motion. We discuss this error recovery in Sec. V-C.

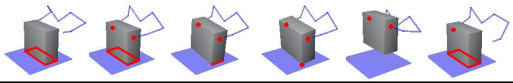
## III. Description of Manipulation Based on a Contact States Graph

In this section, we define the contact states used in description and introduce to describe object manipulation as a CSG.

### A. Definitions of Contact States used in Description

We employ an object-robot contact (OR contact) and an object-environment contact (OE contact) as a description of manipulation. We define an OR contact considering all contacts to be used. We represent an OR contact by three states: all contact points are On (all-on), all contact points are Off (all-off), and some contact points are Off (some-off). For example, in the case of dual-armed manipulation, all-on is that both hands have contacts, all-off is that both hands have no contacts, and some-off is that either hand has no contact. We also define an OE contact based on the number of OE contact points. We represent an OE contact by four states: no-contact, point-contact, line-contact, and face-contact. We do not use robot-environment contacts (RE contact) as a description of manipulation because they do not include contact information of objects and RE contacts have a relationship with the humanoid's self locomotion.

We define contact states used in description by integrating OR contacts and OE contacts. Using OE contacts is a straightforward method to describe object's contact transition. Here, we integrate OR contacts with OE contacts to treat transition of both contacts uniformly. We employ six contact states as Fig.2. In "Separate" state (sep), OE contact is face-contact and OR contact is all-off. In "Double-contact" states, the object has contact both with the robot and the environment. "Double-contact-face" (dbl-f), "Double-contact-line" (dbl-l), and "Double-contact-point" (dbl-p) correspond to face OE contact, line OE contact, and point OE contact. In "Coalition" state (co), OE contact is no-contact and OR contact is all-on. In "Contact-Error" state (err), OR contact is some-off.



| **Contact States** (abbr) | Separate (sep) | Double-contact-face (dbl-f) | Double-contact-line (dbl-l) | Double-contact-point (dbl-p) | Coalition (co) | Contact Error (err) |
|---|---|---|---|---|---|---|
| **OE Contact** | Face | Face | Line | Point | No | * |
| **OR Contact** | All-off | All-on | All-on | All-on | All-on | Some-off |

Fig. 2. Definitions of Contact States
Contacts (red), objects (gray), environments (blue), and robots (green)

### B. Description of Manipulation as a Contact States Graph

We describe contact states transition by using a CSG. A graph corresponds to manipulation of an object. In the execution of manipulation, the system executes the humanoid's motion by tracing each node and switching Manipulation Strategies.

We define a phase during manipulation according to the six contact states in Fig.2. In a CSG, nodes are phases and arcs are transitions. We classify phases into two types as well as ($\alpha$) and ($\beta$): transition phases in which the contact

states change and steady phases in which they do not change. In this way, we can treat convergence of Objective Motion and contact states transition in the same way. For example, we define a phase in "co" contact state as a "co-manip" phase and a phase switching "dbl-f" to "co" as a "dbl-f→co" phase.

### C. Generating a Contact States Graph

Although we can define a graph by connecting nodes with feasible arcs by manual, here we introduce a method to generate a graph from simple descriptions and heuristics. This corresponds to planning of contact states transition. Fig.3 shows three types of generation and heuristics. We distinguish phases which have the same names by adding indices for descriptive purposes: e.g., "sep1-manip" is "sep-manip" in reaching motion and "sep2-manip" is in releasing motion.

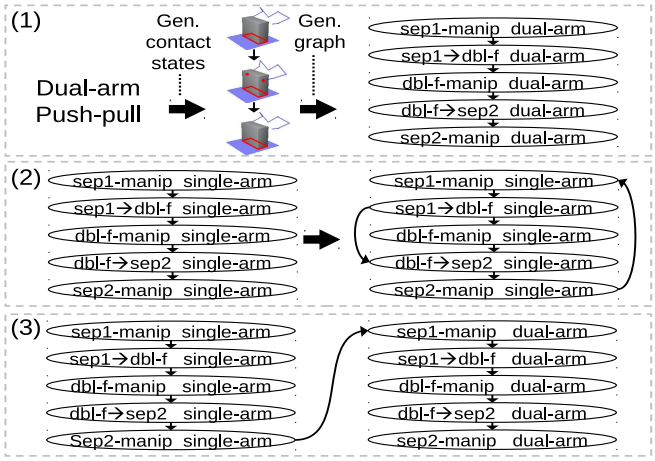



Fig. 3. Connection of Arcs and Heuristics
(1) : Connect based on manipulation primitive
(2) : Releasing and reaching heuristics
(3) : Multi-primitive heuristics

Fig.3 (1) shows generation of a graph based on a manipulation primitive, which is symbolic information specified by the number of OR contacts and the type of manipulation. We specify the number like "single-arm manipulation". The type is symbolic representation including time-series contact states. In this paper, we use "push-pull" (sep1, dbl-f, and sep2), "lift-up" (sep1, dbl-f1, co, dbl-f2, and sep2), "tumble" (sep1, dbl-f1, dbl-l, dbl-f2, and sep2), and "pivot" (sep1, dbl-f1, dbl-l1, dbl-p, dbl-l2, dbl-f2, and sep2, or sep1, dbl-f1, dbl-p, dbl-f2, and sep2). Fig.3 (1) is an example of generation of a graph from a "dual-arm push-pull" primitive.

Fig.3 (2) shows a heuristic to recover from manipulation errors. If the system detects a manipulation error, it executes a reaching motion again after executing a releasing motion. We implement this recovery by connecting the phase in which the system can detect the error to "dbl-f2→sep2" phase and "sep2-manip" phase to "sep1-manip" phase. Fig.3 (2) is an example of connection for recovery from grasping error in "single-arm push-pull" manipulation.

Fig.3 (3) shows a heuristic to utilize multiple manipulation primitives. If several primitives are prepared, the system connects phases of each primitive based on a priority. We implement this by connecting "sep2-manip" of a primitive to "sep1-manip" of the other primitive. Fig.3 (3) is an example to connect "single-arm push-pull" and "dual-arm push-pull" by prioritizing single-armed manipulation over dual-armed manipulation.

## IV. Manipulation Strategies for Each Phase

### A. Manipulation Strategy

Our proposed system switches Manipulation Strategies corresponding to phases. We define a Manipulation Strategy as a set of Manip. Info, Transition Condition, Motion Generator, and Objective Motion. We can use the same Manipulation Strategy if the phase is the same. Fig.5 shows controller substantiation by adding inputs for the Contact-force Controller based on a Manipulation Strategy.

Objective Motions is a reference trajectory, position and orientation for an object required by the Motion Generator. Manip. Info is the object's information that the Motion Generator requires in calculating humanoid's motion based on Objective Motion. Based on Manip. Info, we describe estimators for object's mass properties, desired operational force, and actual object motion. Transition Condition is a condition sentence for phase transition. Because the system knows the destinations for transition from an arc of a CSG, the programmer just implements each condition. By default, Transition Condition in the steady phases observes convergence of Objective Motion and that in the transition phases detects changes of contact states. The Motion Generator generates hands and feet trajectories and key-pose full-body posture sequences based on Objective Motion.

### B. Manipulation Strategies for Each Phase

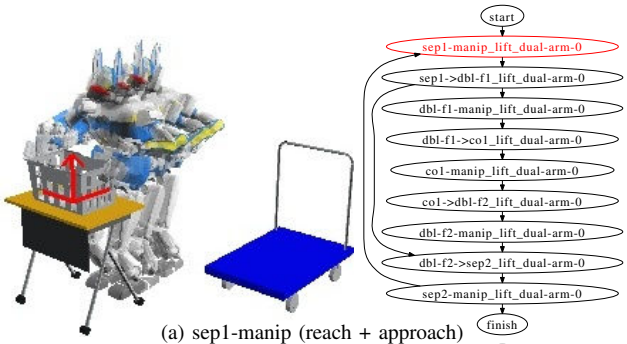
We introduce detailed descriptions of Manipulation Strategies.

"sep-manip" phases correspond to reaching and releasing motions. We implement the Motion Generator by using a footstep planner to approach objects and a full-body inverse kinematics solver for reaching and releasing motions. Manip. Info is the position and orientation of objects. We employ visual recognition for an estimator of Manip. Info.

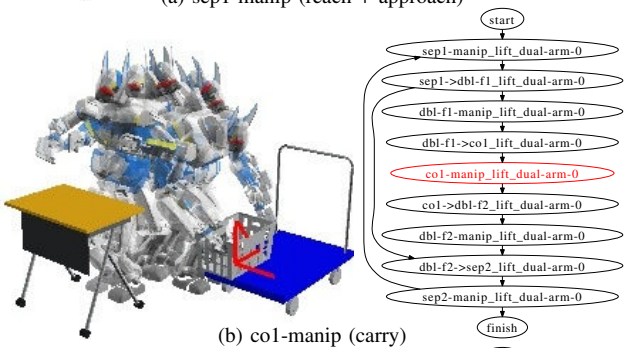
The transition phases between "sep" and "dbl-f" correspond to appearance and disappearance of OR contacts such as grasping by the hands or contacts without hand grasping. Motion Generator is generator for grasping motion or contact force generation. We implement Transition Condition as detection of grasping success based on measurement of hand joint angles for hand grasping. In the case of OR contact other than hand grasping, we implement the condition as detection of contacts based on measurement of OR contact force.

The transition phases between "dbl-f" and "co" correspond to appearance and disappearance of OE contact. Here, we call the appearance rising and the disappearance landing. Transition Conditions are detection of rising and landing. The system detects rising by detecting saturation of estimated OE reaction force[10].

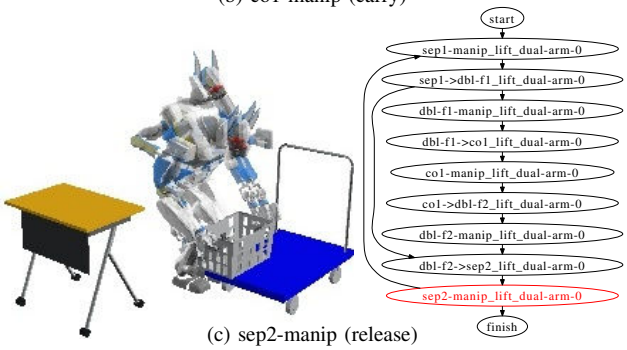
In “co-manip” phase, an object has contacts only with a humanoid robot. We employ an object’s mass properties and an estimator for them as Manip. Info. By adding estimated mass properties to the humanoid’s dynamics model, we can use the same Motion Generator prepared for a humanoid robot itself and consider the acceleration of the object.




(a) sep1-manip (reach + approach)




(b) co1-manip (carry)




(c) sep2-manip (release)

Fig. 4. Motion Planning based on Objective Motion for Each Phase
Left pictures : Planned motion for carrying a basket. Red arrows are Objective Motion. For (c), the ending posture is Objective Motion.
Right graphs : Current phases (red ellipses) in a “dual-arm lift” CSG

The transition phases and manipulating phases in “dbl-f”, “dbl-l”, and “dbl-p” correspond to graspless manipulation [11]. We employ necessary operational forces and actual object motion as Manip. Info. We also employ Motion Generators specific for graspless manipulation.

Fig.4 shows the example of motion planning for basket carrying based on Manipulation Strategy. Fig.4 is the results of simulation so that we disabled sensor-feedback functions such as estimation of object’s mass or forces and OR and OE contact observer. We planned a CSG from “dual-arm lift” primitive (the right sides of Fig.4(a)-(c)). The left side of Fig.4(a)-(c) correspond to planned motion. Fig.4(a) is motion planning in “sep1-manip” phase. In this case, as Motion Generator, the system utilizes footstep planner for approaching and plans reaching motion by solving full-body inverse kinematics. The red arrow shows the target basket position and orientation as the Objective Motion. Fig.4(b) is motion planning in “co1-manip” phase. In this case, as Motion Generator, the system plans lifting-up motion at the first standing point, footsteps to move to the cart, and put-down motion besides the cart. The red arrow also shows the target basket position and orientation as the Objective Motion. Fig.4(c) is motion planning in “sep2-manip” phase. In this case, as Motion Generator, the system plans releasing motion by solving full-body inverse kinematics as “sep1-manip” phase. The Objective Motion is the target joint angles at the end of releasing motion.

## V. Autonomous Execution of Manipulation and Error Recovery

In this section, we introduce a method to execute humanoid’s manipulation based on a CSG and Manipulation Strategies. Moreover, we discuss error recovery using our proposed system.

### A. Execution of Manipulation

The system executes manipulation by using a CSG and Manipulation Strategies as following procedures:

(1) Trace nodes in a CSG : The system traces the CSG. Expediently, we define the first phase as “start” and the final phase as “final”.
(2) Switch controller parameters : According to the current phase, the system switches Manipulation Strategies.
(3) Execute motion : The system executes humanoid’s motion based on Manipulation Strategies. It generates humanoid’s motion satisfying Objective Motion by using Motion Generator. The Contact-force Controller controls humanoid’s balance and contact forces. This execution enables the humanoid robot to utilize adequate controller parameters according to the actual contact states. If the system detected that transition condition is satisfied, it returns to (1) and traces nodes again.

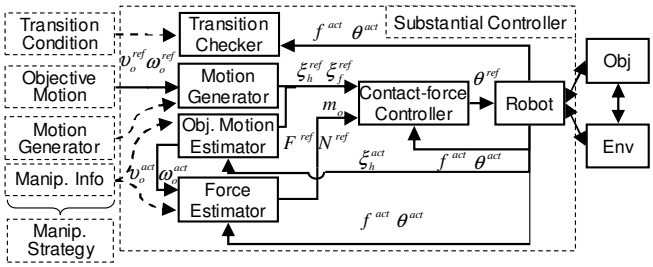



Fig. 5. Substantiate Contact-force Controller based on a Manipulation Strategy

### B. Contact-force Controller for Motion Execution

We utilize a single Contact-force Controller by switching inputs to execute motion at each phase. The Contact-force Controller in Fig.5 requires hands and feet trajectories, reference forces at the hands, and estimated object mass

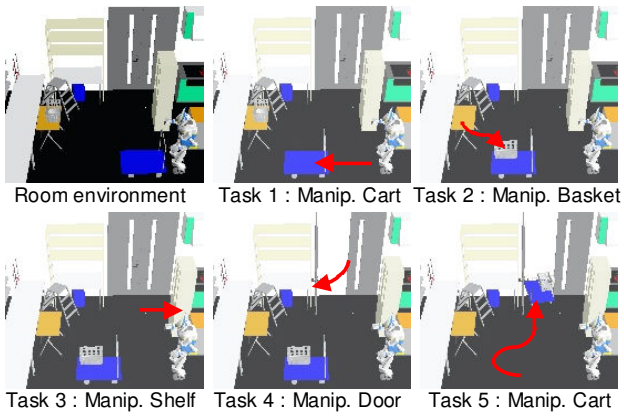



Fig. 6. All Tasks for Carrying Objects
Pictures : Shapes and kinematics models of environment, objects, and robots

properties. It calculates a full-body posture sequence satisfying these inputs. We formulated these calculation in [12], [7]. The Contact-force Controller includes four modules: (1) hands impedance controller, (2) footstep modifier, (3) humanoid's COM trajectory generator, (4) full-body posture sequence generator, and (5) walking stabilizer. The module (2) modifies footsteps based on hand modification to follow the actual object's motion [7]. In (3), we employ Zero Moment Point definition considering hands' reaction forces [13]. The module (3) calculates COM trajectory from hands' reference forces and hands and feet trajectories based on Preview Control [14]. The module (4) calculates full-body posture sequence[15] satisfying hands, feet, and COM trajectories. Contact-force Controller in this paper uses an approximate approach to represent OR and RE contact states. In the case of applying our proposed system to manipulation with complicated RE contacts such as climbing ladders, we need stricter representation of RE contact states like [16].

### C. Error Recovery

A robot possibly fails manipulation if the actual contact state differs from the planned one. In this paper, we call the case just an error and discuss recovery from the error in our proposed system. The CSGs correspond to planned transition of contact states.

We classify error recovery into four types according to whether the system is able to continue manipulation without modifying Objective Motion, needs to modify Objective Motion, needs to switch phases, or needs to switch primitives.

(a) Adaptation by force control: If influence of the error of contact states is small, the system continues manipulation without modifying Objective Motion, phase, or primitives. Contact-force Controller adaptively modifies trajectories of the hands and the feet based on the force errors. Hence the system needs not to modify Objective Motion for the force controller. If the actual contact states are complicated and we cannot classify it into Fig.2, an error of contact states occurs because of a modeling error. In this case, we assume that OE contact is face-contact approximately and we apply "push-pull" primitives. Hence the object is static if the robot has no contact with it and the robot can manipulate it by adapting the actual object motion by using force control. For example, in the case of drawer-pulling, contact between a drawer and a shelf is not face-contact. In this case, however, we can apply the same controller as a pushing manipulation [7].

(b) Re-trying: The system supports error recovery by trying the same action again. We employ two types of re-trying: (b-1) modifying Objective Motion without switching of CSG nodes and (b-2) switching CSG nodes. For an example of (b-1), if an object collides with a wall while pushing, the system should avoid the wall. We can implement this recovery by modifying Objective Motion. For an example of (b-2), if a robot fails to grasp an object, the system can recover from the grasping error by re-grasping. We can implement this recovery by connecting of arcs of the graph.

(c) Switching Primitives: Switching primitives is useful to recover from contact states error in which the robot should change the type of manipulation or the contact points. For example, it is difficult for the robot to continue pushing manipulation if an object starts to tumble. Object's tumbling is determined by mass, COM, and friction so that re-trying as (b) is ineffective. In this case, the system should switch primitives. In detail, the system should change contact points to bring down grasping points or the type of manipulation which does not use pushing against friction forces such as pivoting manipulation or tumbling manipulation. Moreover, this primitive switching is applicable to select manipulation based on evaluation of joint torque[10]. For example, if joint overload becomes large while executing single-armed manipulation, the system should switch to more powerful manipulation such as dual-armed manipulation.

## VI. Objects Carrying Experiment

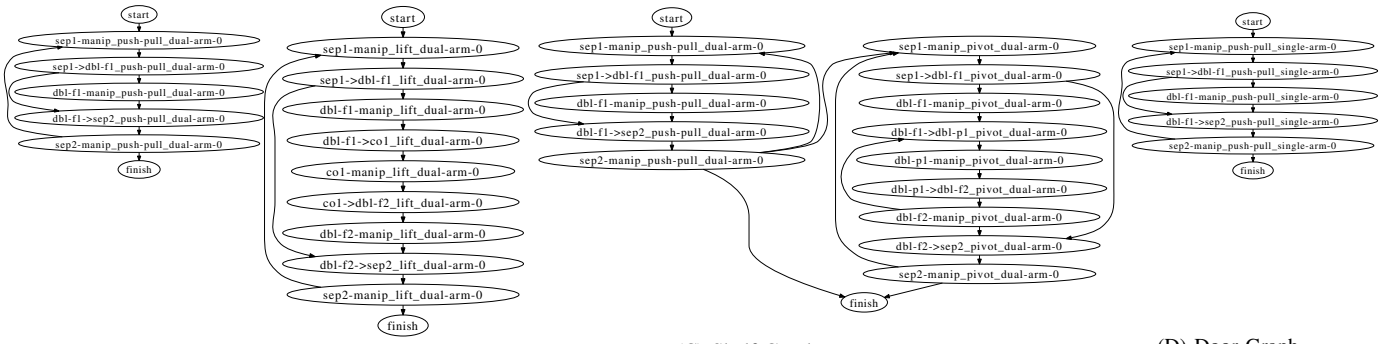
We performed an experiment to evaluate our proposed system. In the experiment, a humanoid robot manipulated four different objects: a cart, a basket, a shelf, and a door. In the experiment, the system performed manipulation without preliminary mass and force information. We used HRP2-JSKNTS with multi-fingered hands, which is the extension of HRP2-JSK[17].

### A. Experimental Conditions and Implementation

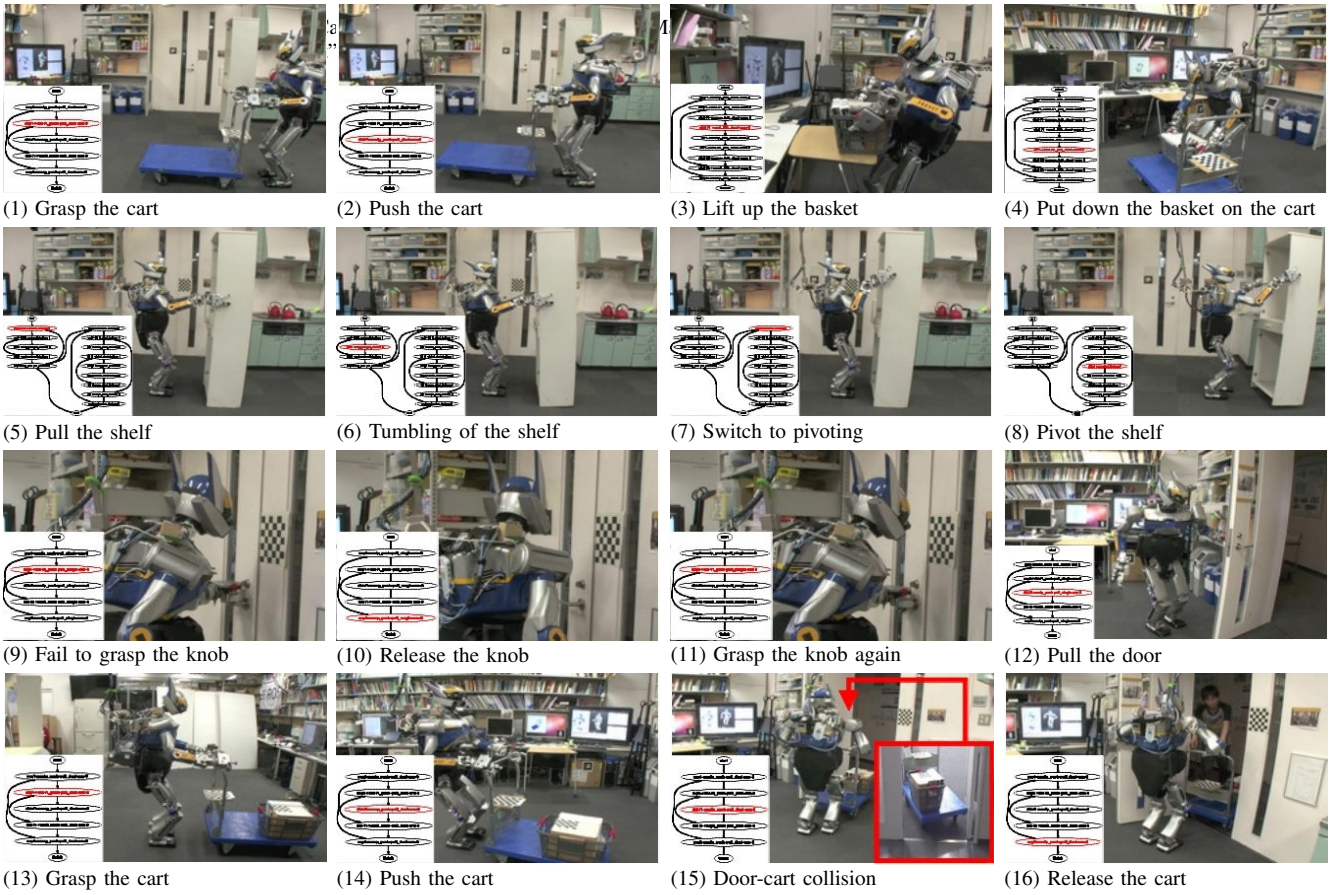
To implement our system the following are required:

- Implementation
  We implement Contact-force Controller. We also implement Motion Generator, and estimator for Manip. Info. for each phase. Once we implement these, we can reuse these implemented controllers and functions if environments or objects change.
- Description
  If we determine the target environment and object, we describe manipulation primitives for ($\alpha$) and Objective Motion for ($\beta$).

For implementation, the system held the shape and kinematics information for the objects and the environment like Fig.6. On the other hand, the estimators specified by Manip.




(A) Cart Graph ("dual-arm push-pull")
(B) Basket Graph ("dual-arm lift-up")
(C) Shelf Graph ("dual-arm push-pull" and "dual-arm pivot")
(D) Door Graph ("single-arm push-pull")



(1) Grasp the cart
(2) Push the cart
(3) Lift up the basket
(4) Put down the basket on the cart
(5) Pull the shelf
(6) Tumbling of the shelf
(7) Switch to pivoting
(8) Pivot the shelf
(9) Fail to grasp the knob
(10) Release the knob
(11) Grasp the knob again
(12) Pull the door
(13) Grasp the cart
(14) Push the cart
(15) Door-cart collision
(16) Release the cart

Fig. 8. Manipulation of Four Objects by Estimating an Unknown Weight and Operational Forces
In all pictures, we depicted graphs at the bottom left in which the current phase are shown as red ellipses.
Task 1 : (1) - (2) : Pushing the empty cart (see Fig.7 (A))
Task 2 : (3) - (4) : Carrying the basket (see Fig.7 (B))
Task 3 : (5) - (8) : Carrying the shelf (see Fig.7 (C))
Task 4 : (9) - (12) : Opening the door (see Fig.7 (D))
Task 5 : (13) - (16) : Pushing the cart with the basket (see Fig.7 (A))

Info estimated the parameters such as the objects' mass properties and necessary operational forces. They utilized measurements of F/T sensors at the hands for OR reaction forces and encoder values for actual motion of the objects. For pivoting manipulation, we prepared pivoting motion generator. We added convergence check of the object's position and orientation to Transition Condition of "dbl-f2-manip" phase in pivoting manipulation. If the object's position and orientation reach the desired values, the system switches to "dbl-f2→sep2" phase. Otherwise, the system returns to "dbl-f1→dbl-p1" phase to continue pivotting. In "dbl-f-manip" for pushing, we commanded the cart velocity by using a joystick. We defined a detector of shelf tumbling as the Transition Condition of the phases of "dual-arm push-pull" primitive in order to transit to "dual-arm pivot" primitive. The detector estimated the shelf inclination based on the hands' orientations by assuming that hand-shelf slipping is small.

For description of the whole task, we programmed the order of each object manipulation, such as carrying the empty cart, putting the basket on it, moving the shelf, opening the door, and carrying the cart with the basket to outside of the

room (Task 1-5 in Fig.6).

For description of each task, we specified primitives for each object and the system planned CSGs based on the primitives. We described primitives as follows: “dual-arm push-pull” for the cart, “dual-arm lift-up” for the basket, “dual-arm push-pull” and “dual-arm pivot” for the shelf, and “single-arm push-pull” for the door. We set especially higher priority for “dual-arm push-pull” than “dual-arm pivot” for the shelf. Therefore the system connected the “sep2-manip” phase of the “dual-arm push-pull” primitive with the “sep1-manip” phase of the “dual-arm pivot” primitive. Fig.7 shows the results of the generated graphs.

### B. Experimental Results and Evaluations

*1) Autonomous Execution:* The system executed humanoid’s manipulation based on generated graphs. Fig.8 shows snapshots of the experiment. In all pictures, we depicted graphs at the bottom left of the pictures corresponding to Fig.7. The red ellipses in the all graphs represent the executing manipulation phases. Fig.8 shows that the system switched manipulation phases and the humanoid robot successfully executed each manipulation. From the experiment, we confirmed that our proposed system is useful to describe and execute object manipulations which include various contact state changes.

We also confirmed that our proposed system enabled the humanoid robot to manipulate objects with an unknown mass or operational force by using adequate estimators. In double-contact phases of the cart, the shelf, and the door in which operational forces are unknown or fluctuate, the system executed automatically operational force estimation according to the phase transition. In Task 1, Task 3, and Task 5, the system executed friction force and moment estimation after switching to the “dbl-f-manip” phase (after Fig.8 (1), (5), and (13)). “Cart Res. Force (Task 1)”, “Cart Res. Moment (Task 1)”, “Cart Res. Force (Task 5)”, and “Cart Res. Moment (Task 5)” in Table.I show the estimated cart resultant forces and moments. In Task 3 and Task 4, the system started to execute operational force update after switching to the “dbl-f-manip” phase (afterFig.8 (7) and (11)). In manipulation of the basket in Task 2, the system switched to the “co-manip” phase by detection of rising, estimated the basket’s mass and COM based on hands reaction forces[1], and finally could successfully carry the basket without the humanoid’s falling down. Fig.8 (3) is “dbl-f→co” phase. Fig.9 ($\alpha$) shows the estimated OE reaction force based on the hands’ reaction forces. The system detected rising at 9[s] because of saturation of the OE reaction force. “Basket Mass (Task 2)” in Table.I is estimated basket’s mass. The actual mass was 7.7[kg] and estimation error was $0.818\%$. “Basket COM (Task 2)” in Table.I is estimated basket’s COM represented in the coordinates of the humanoid’s root link.

*2) Evaluation of Error Recovery:* We evaluated error recovery functions of the system in terms of (a) adaptation, (b) re-trying, and (c) switching according to Sec. V-C.

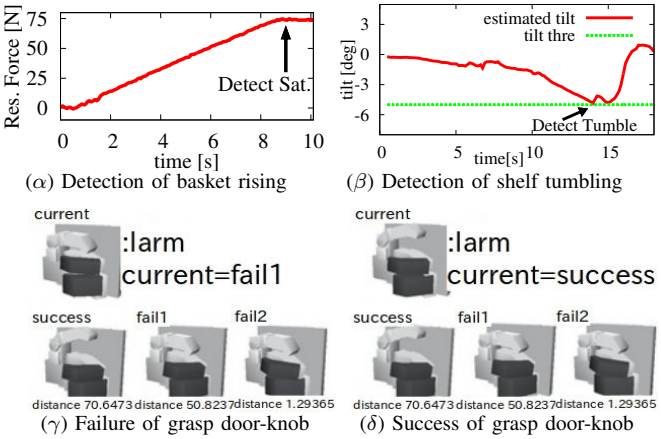
(a) We confirmed that the humanoid robot was able to continue pushing even if the cart collided with the door because of Contact-force Controller. As shown in Fig.8 (15) (see the red rectangle), the cart collided with the door. During pushing, the system assumed that the cart kept contacts only with the ground and the door-ground contact is face-contact so that the door-cart contact was unexpected contact states change. By using Contact-force Controller, the hands motion of the humanoid robot become compliant so that the cart motion also become compliant. First, the cart adapted to the door, and then the humanoid robot followed the actual cart motion by modifying its footsteps according to hand trajectory modification. In Fig.8 (15), Objective Motion of the cart was $0.180$[m/s] forward. The Contact-force Controller successfully adapted footsteps in translation and rotation without modifying Objective Motion.

(b) We confirmed the humanoid robot was able to recover from grasping error by grasping again. In the experiment, we artificially added disturbance before Fig.8 (9) to evaluate re-grasping. In detail, we commanded a grasping motion earlier than adequate timing before Fig.8 (9). Due to this disturbance, the humanoid robot failed to grasp the knob. Therefore the system detected failure of grasping in Fig.8 (9). The system executed a releasing motion in Fig.8 (10) and then a reaching and grasping motion in Fig.8 (11). Fig.9 ($\gamma$) and ($\delta$) show the results of detection OR contacts. The

TABLE I

ESTIMATED OBJECTS’ FORCES, MOMENTS, AND A MASS

| Estimate Target | Value |
|---|---|
| Cart Res. Force (Task 1) | 7.498[N] |
| Cart Res. Moment (Task 1) | 0.790[Nm] |
| Basket Mass (Task 2) | 7.637[kg] |
| Basket COM (Task 2) | $[0.3250 \quad 0.003875 \quad 0.1007]^T$[m] |
| Cart Res. Force (Task 5) | 11.565[N] |
| Cart Res. Moment (Task 5) | 8.859[Nm] |




($\alpha$) Detection of basket rising

($\beta$) Detection of shelf tumbling

($\gamma$) Failure of grasp door-knob

($\delta$) Success of grasp door-knob

Fig. 9. Detection of Phase Transition and Contact State Change
($\alpha$) : Detect rising of the basket from saturation of resultant force at 9[s]. ($\alpha$) corresponds to Fig.8 (3).
($\beta$) : Detect tumbling of the shelf by thresholding at 13.5[s]. ($\beta$) corresponds to Fig.8 (6).
($\gamma$) and ($\delta$): Detect failure and success of grasping ($\gamma$) and ($\delta$) correspond to Fig.8 (9) and (11). Top pictures (“current”) show current hand joint angles. Bottom pictures (“success”, “fail1”, and “fail2”) show reference joint angles for a success case and two failure cases.

detector of grasping error evaluated the distance between the current hand's joint angles and reference joint angles, which are the hand's joint angles for a success case and two failure cases. The bottom three pictures in ($\gamma$) and ($\delta$) correspond to the reference joint angles which we set in advance. In ($\gamma$) corresponding to Fig.8 (9), the distance of "fail1" is the lowest and the system determined that the current grasp was "fail1". Therefore the system preformed releasing and re-grasping. In ($\delta$) corresponding to Fig.8 (11), the distance of "success" is the lowest and the system determined that the current grasp was "success". Therefore the system switched to "dbl-f-manip" phase. In Fig.8 (12), the humanoid robot successfully achieved a door-opening manipulation.

(c) We confirmed that switching of primitives is applicable to recover from manipulation error. In the experiments, the system executed "dual-arm push-pull" in Fig.8 (5). In Fig.8 (6), the shelf started to tumble. Fig.9 ($\beta$) shows the result of the estimator of shelf inclination. The estimated value (the red line) exceeded the threshold (the green line) so that the estimator detected tumbling. After that, the system switched to the "dual-arm pivot" primitive in Fig.8 (7) and completed pivoting manipulation in Fig.8 (8).

## VII. CONCLUSIONS

In this paper, we proposed a uniform method to describe and execute object manipulations for a humanoid robot by focusing on contact states between an object, a robot, and an environment. In Sec. III, we defined contact states based on object-robot contacts and object-environment contacts and proposed a method to describe the contact-transition parts as contact states graphs and the contact-steady parts as objective motion. We also introduced a method to plan CSGs based on manipulation primitives. In Sec. IV, we defined Manipulation Strategies for each phase. We showed that we can reuse the same Manipulation Strategies according to the phases. In Sec. V, we introduced substantiation of the humanoid's controller by switching inputs to Contact-force Controller to achieve manipulation according to the current contact states . We also discussed autonomous execution of manipulation based on actual contact states observation and error recovery in our proposed system. In Sec. VI, we confirmed effectiveness of our proposed method through experiments in which a humanoid robot adaptively manipulated four different objects including various contact state changes. In the experiments, the robot was able to manipulate objects without accurate previous knowledge about object's masses or necessary operational forces because of autonomous execution using CSGs.

## REFERENCES


[1] Kensuke Harada, Shuuji Kajita, Hajime Saito, Mitsuharu Morisawa, Fumio Kanehiro, Kiyoshi Fujiwara, Kenji Kaneko, and Hirohisa Hirukawa. A Humanoid Robot Carrying a Heavy Object. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 1724 – 1729, April, 2005.

[2] Y. Ohmura and Y. Kuniyoshi. Humanoid robot which can lift a 30kg box by whole body contact and tactile feedback. In *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'07)*, pp. 1136–1141, 2007.

[3] T. Takubo, K. Inoue, and T. Arai. Pushing an object considering the hand reflect forces by humanoid robot in dynamic walking. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 1718–1723, 2005.

[4] Mike Stilman, Koichi Nishiwaki, and Satoshi Kagami. Humanoid Teleoperation for Whole Body Manipulation. In *Proceedings of the 2008 IEEE International Conference on Robotics and Automation*, pp. 3175–3180, May 2008.

[5] E. Yoshida, P. Blazevic, V. Hugel, K. Yokoi, and K. Harada. Pivoting a large object: whole-body manipulation by a humanoid robot. *Applied Bionics and Biomechanics*, Vol. 3, No. 3, pp. 227–235, 2006.

[6] Hitoshi Arisumi, Jean-Remy Chardonne, and Kazuhito Yoko. Whole-body motion of a Humanoid robot for passing through a door- Opening a door by impulsive force -. In *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'09)*, pp. 428–434, (2009).

[7] Shunichi Nozawa, Iori Kumagai, Yohei Kakiuchi, Kei Okada, and Masayuki Inaba. Humanoid full-body controller adapting constraints in structured objects through updating task-level reference force. In *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'12)*, pp. 3417–3424, 10 2012.

[8] Francois Keith, Nicolas Mansard, Sylvain Miossec, and Abderrahmane Kheddar. Optimization of Tasks Warping and Scheduling for Smooth Sequencing of Robotic Actions. In *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'09)*, pp. 1609–1614, 2009.

[9] F. Kanehiro, M. Inaba, H. Inoue, and S. Hirai. Developmental Realization of Whole-Body Humanoid Behaviors Based on StateNet Architecture Containing Error Recovery Functions. In *Proc. of International Conference on Humanoids 2000*, 2000.

[10] Shunichi Nozawa, Ryohei Ueda, Yohei Kakiuchi, Kei Okada, and Masayuki Inaba. Sensor-based integration of full-body object manipulation based on strategy selection in a life-sized humanoid robot. *Journal of Robotics and Mechatronics*, Vol. 23, No. 2, pp. 239–248, 2011.

[11] Y. Aiyama, M. Inaba, and H. Inoue. Pivoting: A new method of graspless manipulation of object by robot fingers. In *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'93)*, pp. 136–143, 1993.

[12] Shunichi Nozawa, Yohei Kakiuchi, Kei Okada, and Masayuki Inaba. Controlling the planar motion of a heavy object by pushing with a humanoid robot using dual-arm force control. In *Proceedings of the 2012 IEEE International Conference on Robotics and Automation*, pp. 1428–1435, 5 2012.

[13] K. Harada, S. Kajita, K.Kaneko, and H.Hirukawa. Zmp analysis for arm/leg coordination. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'03)*, pp. 75–81, 2003.

[14] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kensuke Harada, Kazuhito Yokoi, and Hirohisa Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, pp. 1620–1626, Sep 2003.

[15] S.Kajita, F.Kanehiro, K.Kaneko, K.Fujiwara, K.Harada, K.Yokoi, and H.Hirukawa. Resolved Momentum Control: Humanoid Motion Planning based on the Linear and Angular Momentum. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'03)*, pp. 1644–1650, October, 2003.

[16] Hirohisa Hirukawa, Shizuko Hattori, Kensuke Harada, Shuuji Kajita, Kenji Kaneko, Fumio Kanehiro, Kiyoshi Fujiwara, and Mitsuharu Morisawa. A universal stability criterion of the foot contact of legged robots - adios zmp. In *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*, pp. 1976 – 1983, 2006.

[17] Kei Okada, Takashi Ogura, Atsushi Haneda, Junya Fujimoto, Fabien Gravot, and Masayuki Inaba. Humanoid Motion Generation System on HRP2-JSK for Daily Life Environment. In *International Conference on Mechatronics and Automation*, pp. 1772 – 1777, July, 2005.