Grasp planning taking into account the external wrenches acting on the grasped object

Dawid Seredyński and Tomasz Winiarski and Konrad Banachowicz and Cezary Zieliński*

***Abstract*— The paper presents an outline of the specification of a robot controller used in manipulation tasks requiring object grasping by a multi-fingered gripper and interaction with the environment. The specification assumes that the robot is represented as an embodied agent composed of real and virtual effectors and receptors and the control subsystem. The actions of those subsystems are defined by finite state machines invoking in each of their states appropriate behaviours. The presentation focuses on grasp planning and execution.**

I. INTRODUCTION

Grasping is of paramount importance to service robots as mainly through grasping they interact with the environment. Success of a grasp depends not only on grasp planning, but also involves whole robot body motion planning, taking into account the environment model and task properties. Different tasks involving the same object may require different grasps. Both the desired trajectory and the exerted forces during the task execution have be taken into account. This paper presents a structure of an embodied agent [1] able to plan and execute a task specified in terms of a sequence of manipulated object poses and a set of expected external wrenches exerted on the object.

The main contribution of this paper is the grasp planning procedure that takes into account the task specified as a sequence of desired poses of the manipulated object and the expected external wrenches acting on the object during the task execution.

The state of the art is presented in sec. II. Sec. III contains the overall system structure specification. The detailed description of the control subsystem behaviours is presented in sec. IV. Sec. V briefly describes the experiments. The conclusions are drawn in sec. VI.

II. STATE OF THE ART

Using open–source frameworks, such as ROS [2] or Orocos [3], for robot control has become popular, as this shortens the implementation time and enables better code reusability [4], [5]. The specification of complex robotic systems is often based on the overall system structure presented in the form of a data flow diagram [6] with descriptions of algorithms based on the activity diagrams or pseudo code [7]. In the case of component based approach to system design, the system is often divided into parts controlling effectors, receptors and executing specific tasks, including planning routines [6], [8]. The agent based approach has also precipitated into robotics. The concept of an embodied agent consisting of effectors, receptors, both real and virtual, governed by a control subsystem has been utilised to systematically specify diverse robot systems [1], [9], [10]. The actions of each of the mentioned subsystems are represented by a finite state machine (FSM), which in each of its states invokes a behaviour, defined in turn by a transition function and a terminal condition.

Manipulation task planning is a complex problem requiring many planning routines for gripper and manipulator actions [11]. They rely on an environment model and have to invoke collision avoidance algorithms as service robots operate in a human oriented environment that is neither static nor fully predictable. Producing an integrated grasp planner, inverse kinematics solver and manipulator path planner is common. In [11] the planner that plans collision–free trajectory for a given grasp and initial manipulator configuration using RRT search is presented. In [7] the use of precomputed reachability information for solving the inverse kinematics for complex bimanual tasks is presented.

The common approach to grasp planning is the calculation of the Grasp Wrench Space (GWS) using a friction model, for example frictionless point contact (FPC), point contact with friction (PCWF) or soft finger contact with elliptic approximation (SFCE) [12]. The GWS is a set of wrenches that should be resisted by the grasp. It is calculated using the method presented in [13] using either the assumption that the sum magnitude of the contact forces is upper–bounded (L_1 norm) or that the maximum applied finger force is limited (L_∞ norm). The GWS W_{L_1} or W_{L_∞} is represented by the convex hull of the boundary of the GWS in 6–dimensional wrench space. The calculation of the GWS is implemented in Graspit! [14] and Openrave [15]. The quality measure of the grasp based on GWS is usually the distance of the nearest facet of the convex hull from the origin (the worst case disturbance wrench) [13] or the volume of the convex hull [14]. Both of them are task–independent and they do not provide information about the resistance wrench directions. Also, they contain the arbitrarily chosen scaling parameter between force and torque.

When the task is specified, the task oriented grasp planning may be used. The grasping task is usually specified as a set of external wrenches acting on the manipulated object. In [12] tasks are specified by a single wrench, wrench cone or wrench polytope.

As there may be an infinite number of inverse kinematics problem solutions for a redundant manipulator, iterative numerical techniques are used [16], [17] with different

*Institute of Control and Computation Engineering, Warsaw University of Technology, ul. Nowowiejska 15/19, 00–665 Warsaw, Poland, dawid.seredynski@gmail.com

978-1-4799-7043-8/15/$31.00 ©2015 IEEE

heuristics [7], [11], [18]. The IkFast solver is the inverse kinematics solver for redundant manipulators which generates a set of possible solutions. IkFast is numerically stable analytical inverse kinematics solver which uses closed–form solutions [18]. The execution time of the solver is lower than the time needed by other solvers [18], [19]. In the case of a redundant manipulator having 7 degrees of freedom (DOFs), one DOF has to be parametrized in order to make the analytical solution possible for a six dimensional operational space. Such a joint is called a free joint and it is a parameter to the IkFast solver together with the second parameter, the free joint discretization step length (free joint increment). The IkFast searches for inverse kinematics solution for a 6–DOF case with the free joint position set to a value incremented in each step by free joint increment value. The result of the IkFast solver computation is a set of possible arm configurations for the specified end–effector pose.

RRT algorithms are typically used for planning a trajectory in manipulator configuration space [11], [20], [21], e.g. Bidirectional Rapidly–Exploring Random Trees Algorithm (BiRRT) [22]. RRT algorithms are most effective for simple tasks such as single hand manipulation. For more complicated tasks, e.g. closed chain manipulation, optimization–based approach has to be employed [23]. RRT algorithms do not guarantee the generated trajectory to be collision–free and time sampling of the generated trajectory with collision check at each step is required for trajectory validation [24].

III. SYSTEM STRUCTURE

In the conducted experiments the robot was composed of a 7-DOF manipulator, a 3-fingered gripper and an immobile camera. The structure of the embodied agent a_{V} representing this robot is shown in fig. 1. The agent's goal is to plan and execute the specified task.

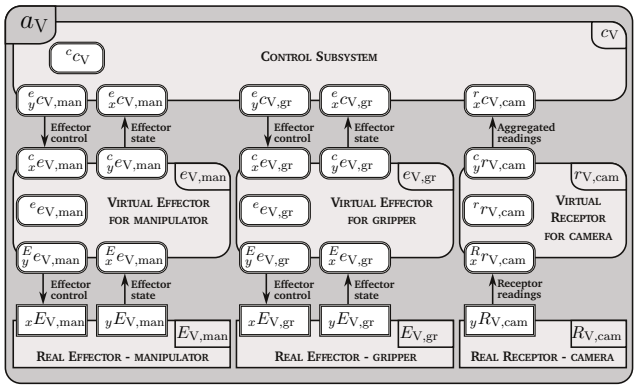

Fig. 1: The structure of the agent a_{V}

A. Real effectors and receptors

Two real effectors were distinguished in the agent a_{V}:

- $E_{\mathrm{V,man}}$ – Kuka LWR manipulator,
- $E_{\mathrm{V,gr}}$ – BarrettHand BH–280 with tactile sensors.

Moreover the agent a_{V} contains a real receptor $R_{\mathrm{V,cam}}$, i.e. a Stand Alone Camera (SAC) [25].

B. Virtual effector $e_{V,man}$

Virtual Effector representing the manipulator is described in [26]. It is responsible for commanding and reading joint positions of the real effector $E_{\mathrm{V,man}}$, i.e. the manipulator. It produces the following behaviours [27]:

- ${}^{e}\mathcal{B}_{\mathrm{V,man,jt}}$ – manipulator motion using joint impedance control with spline interpolation,
- ${}^{e}\mathcal{B}_{\mathrm{V,man,tkex}}$ – task space manipulator motion using Cartesian impedance control with linear interpolation,
- ${}^{e}\mathcal{B}_{\mathrm{V,man,idle}}$ – idle behaviour.

C. Virtual effector $e_{V,gr}$

Virtual Effector representing the gripper is described in [26]. It is responsible for sending commands to and reading joint positions and tactile data (both treated as proprioceptive input) from the real effector $E_{\mathrm{V,gr}}$, i.e. the gripper.

D. Virtual receptor $r_{V,cam}$

In the initial experiments the manipulated objects were recognised and localised by using visual markers – QR codes. The virtual receptor $r_{\mathrm{V,cam}}$ sends the identifiers and poses of the recognized markers to the control subsystem c_{V} [26]. The output buffer from the virtual receptor ${}^{c}_{y}r_{\mathrm{V,cam}}$, connected to the control subsystem input buffer ${}^{r}_{x}c_{\mathrm{V,cam}}$, contains a list of elements $\langle mi, {}^{\mathcal{W}}_{\mathcal{M}_{mi}}\mathcal{T}\rangle$, where mi is the identifier of the marker and ${}^{\mathcal{W}}_{\mathcal{M}_{mi}}\mathcal{T}$ is the pose of the marker mi with respect to (wrt) the world coordinate frame $\mathcal{W}$.

IV. CONTROL SUBSYSTEM c_{V}

The control subsystem reads data from the virtual receptor and virtual effectors, plans the motion of the manipulators and grippers, and sends commands to the virtual effectors to perform the desired actions on the robot and thus the environment. The control subsystem c_{V} is responsible for:

- environment model generation based on the data acquired from the real receptor $R_{\mathrm{V,cam}}$ and processed by the virtual receptor $r_{\mathrm{V,cam}}$ realized by the behaviour ${}^{c}_{+}\mathcal{B}_{\mathrm{V,env}}$ (sec. IV-A),
- grasp and trajectory planning realized by the behaviour ${}^{c}_{+}\mathcal{B}_{\mathrm{V,plan}}$ (sec. IV-C),
- dispatching movement commands to the virtual effectors of: the manipulator $e_{\mathrm{V,man}}$ and the gripper $e_{\mathrm{V,gr}}$ realized by the behaviour ${}^{c}_{+}\mathcal{B}_{\mathrm{V,tkex}}$ (sec. IV-D),
- verification of the stability of the grasp (behaviour ${}^{c}_{+}\mathcal{B}_{\mathrm{V,grver}}$ described in sec. IV-E).

The $+$ subscript in the behaviour symbol indicates that the behaviour terminal condition is tested after the execution of the elementary action based on the transition function. The whole sequence of sensing, planning and acting is realized by the FSM being the core of the control subsystem (fig. 2). Each node of its graph represents a single behaviour.

In the presented system, the task is defined as a motion of the given object caused by the robot in the task–space. It is represented as a list T of homogeneous transformation matrices of the destination poses of the object during the motion and the set W of expected external wrenches acting on the object during the task execution.

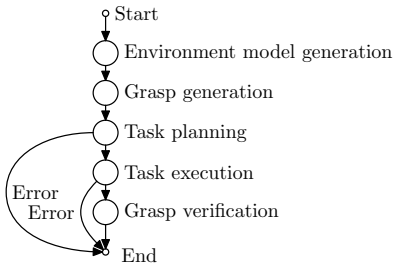

Fig. 2: FSM governing the control subsystem

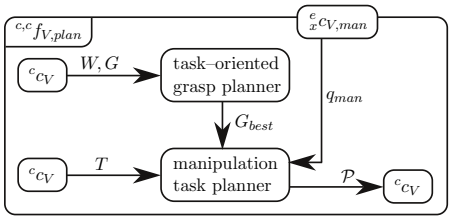

Fig. 3: DFD of the control subsystem transition

A. *Environment model generation* – ${}^{c}_{+}\mathcal{B}_{\text{V,env}}$

It is assumed that all bodies are rigid. Static bodies are used to model such objects as walls, ceiling, floor *etc*. All objects that have an initially unknown pose or a pose that might change are treated as movable bodies.

For each movable object a set of visual markers rigidly attached to it is defined. Each element of the set is a pair $\langle mi, {}^{\mathcal{O}}_{\mathcal{M}_{mi}}\mathcal{T}\rangle$, where mi is the identifier of the marker rigidly connected to the object and ${}^{\mathcal{O}}_{\mathcal{M}_{mi}}\mathcal{T}$ is the homogeneous transformation matrix of the mi marker pose with respect to the object coordinate frame $\mathcal{O}$.

Poses of movable bodies are updated using the absolute poses of visible markers obtained from the buffer ${}^{r}_{x}c_{\text{V,cam}}$ and the information about the relative poses of the markers rigidly connected to the object. The resultant poses of the movable objects are written into the internal memory ${}^{c}c_{\text{V}}$. The result of this behaviour is the geometric model of the environment, that is used by the task planning behaviour ${}^{c}_{+}\mathcal{B}_{\text{V,plan}}$.

B. *Grasp generation* – ${}^{c}_{+}\mathcal{B}_{\text{V,gengr}}$

The set of the generated grasp parameters for the given object is read from a database. The set was generated by simulating closing of the fingers of the gripper in various gripper poses relative to the object using the gripper model and the object model only.

The set of parameters of the generated grasp G is stored in ${}^{c}c_{\text{V}}$. Each element of G is a quadruple $\langle {}^{\mathcal{O}}_{\mathcal{E}}\mathcal{T}, q_{\text{gr,pre}}, q_{\text{gr}}, P\rangle$, where ${}^{\mathcal{O}}_{\mathcal{E}}\mathcal{T}$ is the homogeneous transformation matrix of the end–effector frame $\mathcal{E}$ wrt the object frame $\mathcal{O}$, $q_{\text{gr,pre}}$ is the preshape (initial configuration) of the gripper, q_{gr} is the configuration of gripper at contact and P is the representation of the convex hull of the Grasp Wrench Space (GWS) W_{L_1} boundary generated using the method presented in [13] – it is represented by the set of planes $p \in P$. Each plane is defined as $p = \langle n, d\rangle$, where n is a 6×1 normalized normal vector of the plane pointing outwards from the geometric centre of the convex hull and d is the distance from the origin.

C. *Task planning* – ${}^{c}_{+}\mathcal{B}_{\text{V,plan}}$

Task planning is a two stage process, where the optimal grasp and manipulator configuration for the task execution is selected. The data flow diagram (DFD) of the transition function of this behaviour is shown in fig. 3. The task planning procedure uses two planners:

- task–oriented grasp planner,
- manipulation task planner.

In the first stage the subset of optimal grasps is selected using the task–oriented quality measure. The input argument for the task–oriented grasp planner is a set W of expected external wrenches acting on the object during the task execution. A single external wrench ${}^{\mathcal{O}}w_{\text{ext}} \in W$ acting on the object is the sum of all wrenches that are not exerted by the gripper:

$$ {}^{\mathcal{O}}w = {}^{\mathcal{O}}w_{\text{ext}} + {}^{\mathcal{O}}w_{\text{gr}} $$

where ${}^{\mathcal{O}}w_{\text{gr}}$ is a total wrench exerted by the gripper and ${}^{\mathcal{O}}w$ is a the total wrench acting on the object. All wrenches are represented wrt the object frame $\mathcal{O}$ attached at the centre of mass of the object. The task–oriented grasp planner selects, from the set G_{ev} of evaluated grasps, the subset G_{best} of the best grasps for the specified task. The set of evaluated grasps is calculated as:

$$ G_{\text{ev}} = \mathcal{EW}(G, W) $$

where the function $\mathcal{EW}$ evaluates each grasp and appends the task–oriented quality measure to all grasp descriptions contained in the set G. Thus each quadruple $\langle {}^{\mathcal{O}}_{\mathcal{E}}\mathcal{T}, q_{\text{gr,pre}}, q_{\text{gr}}, P\rangle \in G$ is transformed into quintuple $\langle {}^{\mathcal{O}}_{\mathcal{E}}\mathcal{T}, q_{\text{gr,pre}}, q_{\text{gr}}, P, \mathcal{M}\rangle \in G_{\text{ev}}$. The quality measure of the grasp is

$$ \mathcal{M} = \min_{{}^{\mathcal{O}}w_{ext} \in W} \mathcal{WQ}({}^{\mathcal{O}}w_{\text{ext}}) $$

where the function $\mathcal{WQ}({}^{\mathcal{O}}w_{\text{ext}})$ produces the largest scalar multiplier of the wrench ${}^{\mathcal{O}}w_{\text{ext}}$ keeping the grasp stable:

$$ \mathcal{WQ}({}^{\mathcal{O}}w_{\text{ext}}) = \arg\max_{a \in \mathbb{R}^+}(a \ {}^{\mathcal{O}}w_{\text{ext}} \in W_{L_1}) $$

The quality measure is the largest multiplier of the wrench that has the highest impact on the stability of the grasp.

Let $\mathcal{M}_{\text{best}}$ be the quality of the highest scored grasp in the set G_{ev}. The set G_{best} contains the grasp descriptions for the grasps evaluated above the threshold $\gamma\mathcal{M}_{\text{best}}$, where $\gamma \in [0, 1]$ is a parameter.

In the second stage of the planning procedure, the manipulation plan $\mathcal{P}$ is prepared. The plan evaluated as the best is selected. The manipulation plan is a description of a sequence of actions. Each action is represented as a pair $\langle ActionName, Parameters\rangle$. The possible action types and their parameters are presented in tab. I.

The input argument for the manipulation task planner is a list T of the end–effector poses during the task execution. For each grasp in $\langle {}^{\mathcal{O}}_{\mathcal{E}}\mathcal{T}, q_{\text{gr,pre}}, q_{\text{gr}}, P, \mathcal{M}\rangle \in G_{\text{best}}$ the manipulation plan $\mathcal{P}$ is prepared in the following way:

ActionName	Parameters
MoveJoint	s_q – configuration space trajectory
MoveTask	s_r – task space trajectory
MoveGripper	q_{gr} – desired gripper configuration
Grasp	–

TABLE I: Action types and their parameters. The meaning of actions and their parameters s_q, s_r and q_{gr} is described in sec. IV-D

1) Choose the best manipulator configuration $q_{\text{man,best}}$ from the configuration set generated using the inverse kinematics (IK) solver IkFast for the end effector pose ${}^{\mathcal{W}}_{\mathcal{E}}\mathcal{T} = {}^{\mathcal{W}}_{\mathcal{O}}\mathcal{T}\,{}^{\mathcal{O}}_{\mathcal{E}}\mathcal{T}$. The cost function of the single IK solution increases as the joint positions of the manipulator are closer to their limits and as the wrist configuration is closer to a singular configuration.
2) Set the gripper configuration to grasp preshape $q_{\text{gr,pre}}$ (in simulation). Add action $\langle \text{MoveGripper}, q_{\text{gr,pre}} \rangle$ to the plan $\mathcal{P}$,
3) Plan the joint space trajectory s_q of the manipulator using the BiRRT algorithm, starting from the current position q_{man}, obtained from the buffer ${}^{e}_{x}c_{\text{V,man}}$, and with the destination $q_{\text{man,best}}$. Add action $\langle \text{MoveJoint}, s_q \rangle$ to the plan $\mathcal{P}$,
4) Set the gripper configuration to grasp shape q_{gr} (in simulation). From this step, the object is considered as grasped, i.e. the object is rigidly connected (attached) to the end effector. The object location wrt the end–effector is ${}^{\mathcal{O}}_{\mathcal{E}}\mathcal{T}$. Add actions $\langle \text{MoveGripper}, q_{\text{gr}} \rangle$ and $\langle \text{Grasp}, - \rangle$ to the plan $\mathcal{P}$,
5) Simulate the task–space trajectory s_r of the end–effector for the sequence of end–effector poses obtained from the sequence of the desired object poses from T, starting at the manipulator configuration $q_{\text{man,best}}$. The simulated trajectory of the end–effector is sampled and the inverse kinematics is calculated (closest to the IK solution from the prevoius step) to check if the trajectory can be executed. Add action $\langle \text{MoveTask}, s_r \rangle$ to the plan $\mathcal{P}$.

If any of the above planning steps fails, e.g. a collision with the environment is detected or the IK solution does not exist, then the plan is discarded.

The steps 1 to 3 plan and verify the joint–space trajectory form the start configuration to the grasp configuration and the steps 4 and 5 verify whether the task execution is possible. When the plans for the best grasps in the set G_{best} are prepared, the best plan is selected according to the plan cost function calculated in step 1, i.e. the IK solution cost for the initial manipulator configuration. If no plans could be prepared, the behaviour ends with an error status.

D. Task execution – ${}^{c}_{+}\mathcal{B}_{\text{V,tkex}}$

The prepared plan $\mathcal{P}$ is executed by the real hardware. For each element of the plan, one action is performed:

- MoveJoint – joint–space trajectory execution using joint impedance control: the control subsystem sends the trajectory parameters s_q to the virtual effector $e_{\text{V,man}}$ through the buffer ${}^{e}_{y}c_{\text{V,man}}$, where s_q is a list of elements $\langle q_{\text{jts}}, \dot{q}_{\text{jts}}, t_{q\text{jts}} \rangle$, q_{jts} is a 7×1 vector of joint positions, $\dot{q}_{\text{jts}}$ is a 7×1 vector of joint velocities and $t_{q\text{jts}}$ is the trajectory segment execution duration. The control subsystem waits until the trajectory execution ends.
- MoveTask – task–space trajectory execution: the control subsystem sends the trajectory parameters s_r to the virtual effector $e_{\text{V,man}}$ through the buffer ${}^{e}_{y}c_{\text{V,man}}$, where s_r is a list of elements $\langle r_{\text{tkexs}}, t_{\text{tkexs}} \rangle$, r_{tkexs} is a 7×1 pose vector and $t_{r\text{tkexs}}$ is the trajectory segment execution duration. The control subsystem waits until the trajectory execution ends.
- MoveGripper – changes the gripper configuration: the control subsystem commands the virtual effector $e_{\text{V,gr}}$ to move its joints to the desired value by sending the gripper motion parameters: $\langle q_{\text{grs}}, \dot{q}_{\text{grs}}, \tau_s \rangle$ through the buffer ${}^{e}_{y}c_{\text{V,gr}}$, where q_{grs} are the desired joint positions, $\dot{q}_{\text{grs}}$ are the maximum joint velocities and τ_s are the maximum motor currents. The mentioned maximum values are the limits for the desired values delivered to the closed loop position control; trapezoidal velocity profile is executed by the real effector $E_{\text{V,gr}}$.
- Grasp – reads the tactile pressure vector ψ from the buffer ${}^{e}_{x}c_{\text{V,gr}}$, saves it in the internal memory ${}^{c}c_{\text{V}}$ and verifies whether all fingers have contact with the object. If one or more fingers have no contact, the behaviour ends with an error status.

If during the trajectory execution an error status is obtained through the buffer ${}^{e}_{x}c_{\text{V,man}}$, the behaviour ends with an error status. The virtual effector $e_{\text{V,man}}$ reports an error if the joint position error (in the joint impedance mode) or the end effector pose error (in the Cartesian impedance mode) exceeds the tolerance values stored in the virtual effector internal memory ${}^{e}e_{\text{V,man}}$. Such a situation occurs if the external force acting on the manipulator in impedance control mode is high enough to change the configuration of the manipulator significantly.

E. Grasp verification – ${}^{c}_{+}\mathcal{B}_{\text{V,grver}}$

Grasp verification is performed after the plan is executed successfully. Two tactile readings are compared. The first is obtained just after grasp execution (after execution of the action labelled as Grasp) and the second one at the end of the plan execution. The executed grasp is considered successful if $||\alpha - \beta|| < \mathcal{D}_{max}$, where β is the tactile pressure vector read from the virtual effector buffer ${}^{e}_{x}c_{\text{V,gr}}$ during the ${}^{c}_{+}\mathcal{B}_{\text{V,grver}}$ behaviour, α is the tactile pressure vector saved in ${}^{c}c_{\text{V}}$ during the action Grasp, and $\mathcal{D}_{max}$ is the threshold value.

V. EXPERIMENTS AND IMPLEMENTATION

The experiments[1] were performed on the Velma robot using a redundant 7–DOF arm with three fingered BarrettHand

[1]The footage is available at:
https://robotyka.ia.pw.edu.pl/twiki/bin/view/Robots/Velma

gripper and a camera. The real–time part of the system is implemented as a set of Orocos components supervised by the Xenomai operating system. That includes the virtual effector of the manipulator with closed loop control running at 1 kHz rate. The other parts of the system, without real–time constraints, are implemented as ROS nodes working under Linux. Openrave was used to store the environment model, and for collision detection and trajectory generation. For a cuboid object, two tasks were specified:

- lift–up – the translation along the vertical axis by 0.1 m,
- lift–up with rotation – the sequence of two motions: the translation along the vertical axis by 0.05 m and the same translation combined with rotation about the horizontal axis by 90° wrt the object centre.

Visual markers were used for the initial estimation of the object pose. Due to the large pose uncertainty of the visual markers, the enlarged object models (greater in size) were used for manipulator trajectory planning to avoid hitting obstacles (fig. 4 on the right). At the grasp selection stage, the exact models of objects were used (fig. 4 on the left).

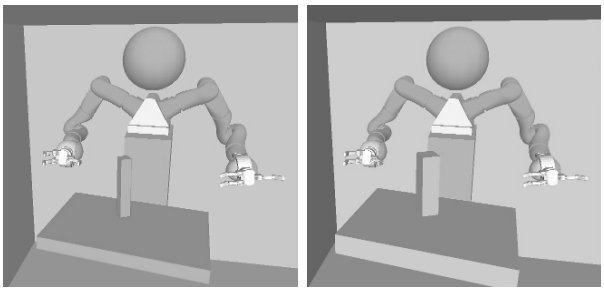

Fig. 4: Environment model used for: grasp planning (left), manipulator trajectory planning with enlarged objects (right)

The set G of grasps for the object was precomputed in simulation and saved in a database. The task was specified as a list T of the desired poses of the object and the set W of expected external wrenches expressed wrt the object frame. The set W was calculated by transforming the expected gravity force acting on the object expressed wrt the world frame into the one wrt the object frame. For each desired pose ${}^{\mathcal{W}}_{\mathcal{O}}\mathcal{T} \in T$ of the object, the expected wrench ${}^{\mathcal{O}}w_{\text{ext}}$ was calculated and upended to the set W:

$$ {}^{\mathcal{O}}w_{\text{ext}} = \begin{bmatrix} \mathcal{A}_{\text{rot}}({}^{\mathcal{W}}_{\mathcal{O}}\mathcal{T}^{-1}) & 0_{3\times3} \\ 0_{3\times3} & \mathcal{A}_{\text{rot}}({}^{\mathcal{W}}_{\mathcal{O}}\mathcal{T}^{-1}) \end{bmatrix} {}^{\mathcal{W}}w_{\text{ext}} $$

where ${}^{\mathcal{W}}w_{\text{ext}} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 \end{bmatrix}$ is the expected wrench related to the force of gravity, the operator $\mathcal{A}_{\text{rot}}$ extracts from a homogeneous transformation matrix a 3×3 rotation matrix and $0_{3\times3}$ is a 3×3 null matrix.

In the grasp selection stage, the task–oriented grasp planner calculates the quality of each grasp for each entry of the database, taking the set W of expected wrenches as the input argument. Grasp evaluation produced different results for each task (fig. 5). For the lift–up task without rotation the majority of grasps had a high quality measure, and for the lift–up task with rotation the grasps more distant from the centre of mass of the object had lower quality measure due to the relatively small external forces and torques they could resist. All grasps evaluated above $0.9\mathcal{M}_{\text{best}}$ were used by the manipulation task planner.

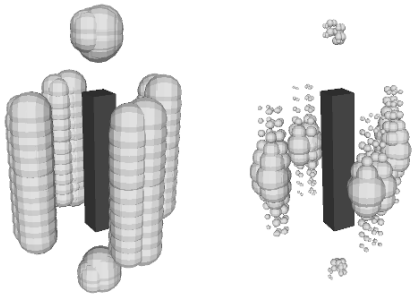

Fig. 5: Visualisation of the task–oriented quality measure for two different tasks: lift–up without rotation (left), lift–up with rotation (right). Each sphere represents the gripper position for a grasp and the size of the sphere represents the quality measure – the larger the sphere, the better the quality of the grasp

In the second stage, the manipulation task planner selected the most convenient configuration for the task execution (fig. 6) and planned the trajectory of the manipulator from the initial pose to the grasping pose. The grasp was selected from the set of grasps suitable for the specified task. For the lift–up task, the top grasp was selected (fig. 6 on the left) with the manipulator configuration penalty 0. For the litf–up and rotation task, the side grasp was selected (fig. 6 on the right) with the manipulator configuration penalty equal to 0.89.

The plan for each task was executed on the Velma robot (fig. 6). For the lift–up and rotation task, the rough estimation of the external wrench acting on the manipulator expressed wrt the tool frame is shown in fig. 7. The external wrench acting on the manipulator was calculated by using the joint torque sensors. The figure shows the rapid rise of the x component of the force as the object is lifted up from the table and the gravity force acting on the object is taken over by the manipulator. During the rotational motion, the x component of force decreases and the z component rises.

The grasp selected for this task can resist the changing force.

VI. CONCLUSIONS

The presented system has the ability to plan the task execution for the task specified as a sequence of desired poses of the manipulated object taking into account the expected external wrenches acting on the object during the task execution. The most suitable grasp for the specified task is selected, i.e the one that has the best possibility of resisting the expected external wrenches.

Future work will include addition of a vision system recognising objects on the basis of RGBD data and implementation of other tasks such as opening a cabinet door and pulling out

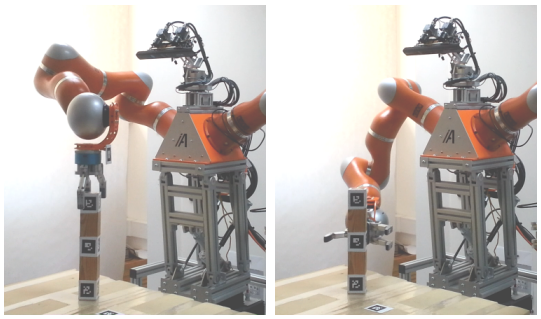

Fig. 6: Grasp selected for the lift–up task without rotation (left), and with rotation (right)

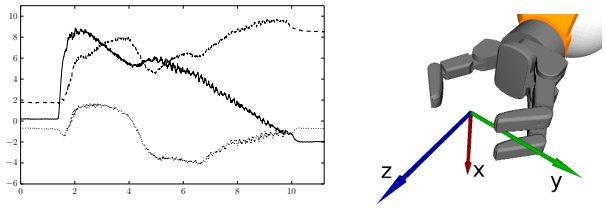

Fig. 7: Plot of forces [N] measured wrt time [s] (left) wrt the tool frame (right): x (solid line), y (dotted), z (dashed), during the motion using Cartesian impedance control: lift–up and rotation of the grasped object.

objects from the cabinet. Also, the grasp stability verification behaviour may be used for gathering the experience for machine learning.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support of this work by The National Centre for Research and Development grant no. PBS1/A3/8/2012.

REFERENCES

[1] C. Zieliński, "A unified formal description of behavioural and deliberative robotic multi-agent systems," in *7th International IFAC Symposium on Robot Control (SYROCO)*, vol. 7, 2003, pp. 479–486.

[2] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System," in *Proceedings of the Open-Source Software workshop at the International Conference on Robotics and Automation (ICRA)*, 2009.

[3] H. Bruyninckx, "The real-time motion control core of the OROCOS project," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, September 2003, pp. 2766–2771.

[4] I. Zubrycki and G. Granosik, "Test setup for multi-finger gripper control based on robot operating system (ros)," in *Robot Motion and Control (RoMoCo), 2013 9th Workshop on*. IEEE, 2013, pp. 135–140.

[5] T. Winiarski and K. Banachowicz, "Opening a door with a redundant impedance controlled robot," *9th Workshop on Robot Motion & Control (RoMoCo)*, pp. 221–226, 2013.

[6] T. Asfour, P. Azad, N. Vahrenkamp, K. Regenstein, A. Bierbaum, K. Welke, J. Schroeder, and R. Dillmann, "Toward humanoid manipulation in human-centred environments," *Robotics and Autonomous Systems*, vol. 56, no. 1, pp. 54–65, 2008.

[7] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Efficient inverse kinematics computation based on reachability analysis," *International Journal of Humanoid Robotics*, vol. 9, no. 04, 2012.

[8] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Şucan, "Towards reliable grasping and manipulation in household environments," in *Experimental Robotics*. Springer, 2014, pp. 241–252.

[9] C. Zieliński and T. Winiarski, "Motion generation in the MRROC++ robot programming framework," *International Journal of Robotics Research*, vol. 29, no. 4, pp. 386–413, 2010.

[10] C. Zieliński, T. Kornuta, and T. Winiarski, "A systematic method of designing control systems for service and field robots," in *19-th IEEE International Conference on Methods and Models in Automation and Robotics, MMAR'2014*. IEEE, pp. 1–14.

[11] D. Bertram, J. Kuffner, R. Dillmann, and T. Asfour, "An integrated approach to inverse kinematics and path planning for redundant manipulators," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. IEEE, 2006, pp. 1874–1879.

[12] R. Haschke, J. J. Steil, I. Steuwer, and H. Ritter, "Task-oriented quality measures for dextrous grasping," in *Computational Intelligence in Robotics and Automation, 2005. CIRA 2005. Proceedings. 2005 IEEE International Symposium on*. IEEE, 2005, pp. 689–694.

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

[14] A. T. Miller and P. K. Allen, "Graspit! a versatile simulator for robotic grasping," *Robotics & Automation Magazine, IEEE*, vol. 11, no. 4, pp. 110–122, 2004.

[15] R. Diankov and J. Kuffner, "Openrave: A planning architecture for autonomous robotics," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, p. 79, 2008.

[16] C. A. Klein and C.-H. Huang, "Review of pseudoinverse control for use with kinematically redundant manipulators," *Systems, Man and Cybernetics, IEEE Transactions on*, no. 2, pp. 245–250, 1983.

[17] L. Sciavicco and B. Siciliano, *Modelling and control of robot manipulators*. Springer Science & Business Media, 2000.

[18] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, 2010.

[19] R. Diankov, K. Sato, H. Yaguchi, K. Okada, and M. Inaba, "Manipulation planning for the jsk kitchen assistant robot using openrave," in *The 29th Annual Conference on Robotics Society of Japan, AC2Q2–2*, 2011.

[20] P. Michel, C. Scheurer, J. Kuffner, N. Vahrenkamp, and R. Dillmann, "Planning for robust execution of humanoid motions using future perceptive capability," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*. IEEE, 2007, pp. 3223–3228.

[21] M. Stilman, "Task constrained motion planning in robot joint space," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*. IEEE, 2007, pp. 3074–3081.

[22] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2. IEEE, 2000, pp. 995–1001.

[23] W. Szynkiewicz and J. Błaszczyk, "Optimization-based approach to path planning for closed-chain robot systems," *International Journal of Applied Mathematics and Computer Science*, vol. 21, no. 4, pp. 659–670, 2011.

[24] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Efficient motion planning for humanoid robots using lazy collision checking and enlarged robot models," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*. IEEE, 2007, pp. 3062–3067.

[25] M. Staniak and C. Zieliński, "Structures of visual servos," *Robotics and Autonomous Systems*, vol. 58, no. 8, pp. 940–954, 2010.

[26] T. Winiarski, K. Banachowicz, and D. Seredyński, "Multi-sensory feedback control in door approaching and opening," in *Intelligent Systems'2014*, ser. Advances in Intelligent Systems and Computing, D. Filev, J. Jabłkowski, J. Kacprzyk, M. Krawczak, I. Popchev, L. Rutkowski, V. Sgurev, E. Sotirova, P. Szynkarczyk, and S. Zadrozny, Eds. Springer International Publishing, 2015, vol. 323, pp. 57–70.

[27] ——, "Two mode impedance control of Velma service robot redundant arm," in *Progress in Automation, Robotics and Measuring Techniques. Vol. 2 Robotics.*, ser. Advances in Intelligent Systems and Computing (AISC), R. Szewczyk, C. Zieliński, and M. Kaliczyńska, Eds., vol. 351. Springer, 2015, pp. 319–328.