Autonomous Bipedal Humanoid Grasping with Base Repositioning and Whole-Body Control

A. M. Sundaram; B. Henze; O. Porges; Z. Marton; M. A. Roa

Autonomous Bipedal Humanoid Grasping with Base Repositioning and Whole-Body Control

Ashok M. Sundaram[1], Bernd Henze[1], Oliver Porges[1], Zoltán-Csaba Márton[1], Máximo A. Roa[1]

Abstract— **Autonomous behaviors in humanoid robots are generally implemented by considering the robot as two separate parts, using the lower body for locomotion and balancing, and the upper body for manipulation actions. This paper provides a unified framework for autonomous grasping with bipedal robots using a compliant whole-body controller. The grasping action is based on parametric grasp planning for unknown objects using shape primitives, which allows a generation of multiple grasp poses on the object. A reachability analysis is used to select the final grasp, and also for triggering a base repositioning behavior that locates the robot on a better position for grasping the desired object more confidently, considering all grasps and the uncertainty in reaching the desired position. The whole-body controller accounts for perturbations at any level and ensures a successful execution of the intended task. The approach is implemented in the humanoid robot TORO, and different experiments demonstrate its robustness and flexibility.**

I. INTRODUCTION

Humanoid robots are intended for performing mobile manipulation in human environments, for replacing humans in dangerous areas, or for taking over physically demanding or repetitive tasks. Autonomous behaviors are expected for successfully executing different types of actions, including walking, multi-contact locomotion, and grasping and manipulation of objects. Due to the large number of degrees of freedom required to create a humanoid robot, motion planning and control of the robot become complex and expensive problems. A common solution adopted in the community is the consideration of lower and upper body as separate parts: the upper body executes a manipulation task, and the lower body uses a balancing controller to keep the robot in equilibrium while considering the manipulation forces as perturbations for the balancer. In typical mobile manipulation scenarios using industrial or humanoid (wheeled) robots, the robot moves to a desired location, stops and freezes the lower body, and then executes the manipulation task [1].

The autonomous robot is expected to decide what is the best location for performing an intended manipulation action, or to reposition its base if a selected base location is not good enough for performing the task (Fig. 1). Different techniques have been proposed for this purpose. Reachability inversion maps encoding the inverse kinematics information are used to compute multiple base locations at the floor level for a wheeled humanoid robot to achieve one desired TCP location (which corresponds to a goal grasp) [2]. This concept was

Fig. 1: Humanoid robot TORO [5] reaching to grasp a ball while actively balancing against the human perturbation.

extended to find the SE(2) stance (2D position and 1D orientation) for a bipedal humanoid robot [3]. A coupling of forward and inverse dynamic reachability maps was proposed in [4] to plan lower body foot positions and upper body arm configurations to do a manipulation task while maintaining stability. All of these methods provide multiple base poses for a single desired TCP location, even though it might be possible to grasp an object with diverse TCP locations.

A recent approach in [6] also uses inverse reachability maps, but accounts for multiple task and grasp poses. It finds all possible base positions from an inverse reachability map that satisfies the task poses, clusters them and assigns success probability scores for each cluster. Clusters with higher scores are again processed to find the best orientation for the base by analyzing the reachability map. Another consideration of multiple goals is through the use of a surface map [7], which allows finding the best robot pose for a specific polygon (for instance, a table) that represents the target workspace rather than a best pose for a target object.

All of the above methods assume that the computed base pose is perfectly reachable using an appropriate path/foot step planner and controller, and do not consider that there will be uncertainties in reaching the base pose precisely. Moreover, they are computed for manually assigned goal grasping poses. One alternative to automatically create a pool of grasp poses is through parametrized grasp planning using geometric primitive shapes. The simplification of grasp planning through primitives was introduced in [8]. A decomposition of the CAD model of the object in primitive shapes [9] or in boxes [10] has also been proposed to create a database of simplified objects that is compared through graph

matching with the primitive shapes detected on the scene. For unknown objects, descriptors based on shape context are used for learning approaches to generate grasps [11].

This paper provides an approach for the generation of autonomous grasping behaviors for unknown objects using a bipedal humanoid robot. The objects are visually recognized and classified as a set of primitive geometries, for online generation of parametrized grasps for the object. When the initially chosen robot pose is not good enough for performing the grasping task, the robot is automatically repositioned based on a reachability analysis. This analysis is performed for a grasping region, which takes into account uncertainties in the positioning of the mobile robot, and simultaneously considers multiple goal grasps, thus avoiding dependency on a single goal grasp or on a discrete set of pre-defined grasps. Finally, the grasping action is executed using a whole-body compliant controller that continuously updates the position of the arm to effectively reach the grasping target, and avoids splitting the control of the humanoid robot into separate threads for the lower and upper body.

II. APPROACH OVERVIEW

The overall architecture of the approach used in this work for autonomous humanoid grasping is shown in Fig. 2. It represents in different colors the blocks that correspond to perception, planning, and control. The intended application is a tabletop scenario, where the object is placed on a flat dominant surface. In our experiments, we used AprilTags to mark different tables. Once the AprilTag is located, the robot walks to a predefined location with respect to the AprilTag. For walking, the robot uses a simple step planner in Cartesian space that moves toward the goal in two stages. In the first stage, the robot approaches an intermediate location while reaching a desired orientation, and in the second stage, the robot moves toward the goal following a straight line, i.e. keeping the desired orientation. The success of the walking motion is ensured through visual odometry and SLAM; our implementation uses ORB-SLAM2 [12]. The components described so far have low complexity (step planner) or are relatively standard (AprilTag detection, SLAM), and will not be explained in further detail in the paper.

After reaching the goal position for entering the manipulation phase, the robot captures data from the scene, and a visual pipeline allows the detection of the object of interest. The object is approximated as a primitive geometrical shape, and that shape is also used for the parametric grasp planning, as described in Section III. The subsequent grasp selection filters out unreachable and colliding grasps, and ranks the remaining grasps according to a desired quality metric. When there are no reachable grasps, the base of the robot must be repositioned so that there is a higher confidence in obtaining a successful grasp. A reachability analysis using Reachability and Capability maps [13] allows the selection of the optimal base position, as presented in Section IV. The optimal position is identified while considering the uncertainty in walking to the goal precisely. After reaching the new base position, the robot repeats the perceive-plan-act cycle, where
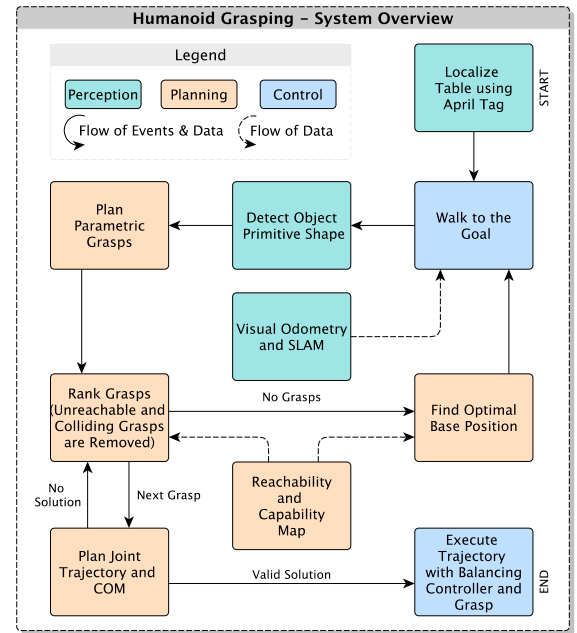

Fig. 2: Overview of the architecture for autonomous bipedal humanoid grasping.

multiple feasible grasps are usually available. In rare cases, if the robot fails to reach the base position within the allowed error threshold used to model the uncertainty, there might be no feasible grasps. To overcome such cases, a new optimal position is recalculated and the cycle is repeated.

When there are feasible grasps available, the next step is the computation of a trajectory in joint space for the kinematic chain for manipulation (right arm plus hip in our case, due to the poor reachability and lack of redundancy of the 6 DoF arms of TORO). Once the path is obtained, it is executed using a whole-body compliant controller. The whole-body balancing controller and its relation with the control of the manipulation chain are discussed in Section V. Finally, Section VI presents different experiments on the humanoid robot TORO [5] that demonstrate the flexibility and robustness of the approach.

III. GRASPING BASED ON PRIMITIVE SHAPES

A human interacts daily with multiple objects of different shapes, but most of these objects (or at least certain parts of them) can be assimilated to one primitive shape. By doing so, the manipulation of these objects can also be simplified by allowing the robot to plan grasps online while considering few constraints (rechability, no collision), and later on, move towards other constraints or actions as required to cope with the uncertainties caused by the simplification of the object. This section describes such vision and grasp synthesis module for primitive shapes.

A. Primitive Shape Fitting

The algorithm used here to automatically decompose the scene into shape primitives was initially introduced in [14]
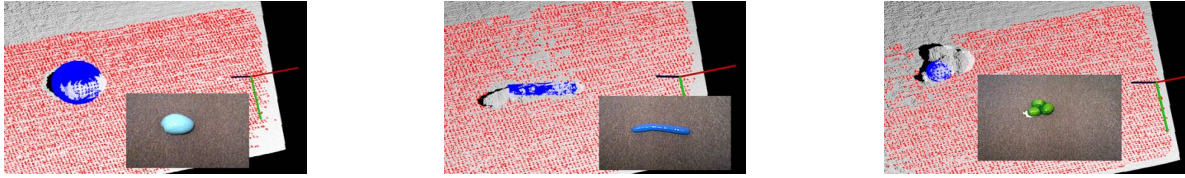
Fig. 3: Parametric shape fitting of different objects. Primitive shape is highlighted in blue. Dominant plane is highlighted in red. Camera frame and the RGB image of the scene are also embedded. From left to right: (i) Mango-like object detected as a sphere, (ii) Cucumber-like object detected as a cylinder, (iii) Individual lime from a bag detected as a sphere.

```
plane "table"
...
volume "tabletop" : outside of "table"
# along surface normal, bounded by contour
...
sphere box cylinder "object" in "tabletop" \\
precluster 3cm 50 # minimum number of points
    BSdiameter: [2cm − 35cm]
    width: [2cm − 30cm] # x axis
    height: [2cm − 30cm] # y axis
    depth: [2cm − 30cm] # z axis
```

Listing 1: Example of scene parsing description to search for a sphere, box, or cylinder on a table.

and extended in [15]. The algorithm was successfully employed for robotic manipulation in [16], however with a fixed set of taught grasps. Starting with depth images or (multi-view) point clouds, it robustly fits different basic shapes into the scene, leaving the unmodeled points as obstacles. Several objects next to or on top of each other can also be handled due to physical plausibility checks. The basic algorithm was complemented with the definition of additional constraints, e.g. regions of interest or maximum size of the primitive, in order to improve the runtime. As the intended application is a tabletop scenario, a dominant planar surface is first searched for and cropped, and then different primitive shapes are fitted to the points lying on top of it. This is achieved using a YAML scene description, as shown in Listing 1.

The initial algorithm in [15] forced the system to search for each possible shape that matches the specific constraints separately. To use the shape fitting in a fully automatic way, the scene parsing language was extended to specify a list of possible shapes, from which the best match to each cluster will be returned. The shape constraints are then limited to the ones that are common to all shapes, namely the bounding sphere diameter (BSdiameter) and the extents. The extents of the shape can be limited such that the detected object is graspable by the hand.

More complicated shapes (e.g. cone, torus, surfaces of revolution) can also be fit, but this work uses three primitive shapes, namely spheres, boxes, and cylinders, as those constitute the vast majority of shape primitives we expect to encounter. Some examples of the shape fitting process on objects that are not perfectly shaped are shown in Fig. 3. The YCB set of objects [17] was used to estimate the proportion of commonly encountered objects to be manipulated that

could be modeled using this approach. From the 72 YCB objects, 41 objects have graspable parts that can be fitted with one of the three shape primitives robustly, 6 objects work only under certain orientations (Fig. 4a), and the remaining 25 objects cannot be modeled (Fig. 4b), although object-specific tuning of parameters and constraints might work in some cases.
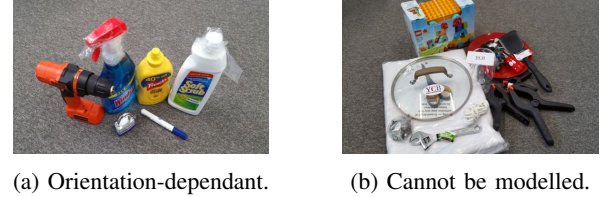
(a) Orientation-dependant.

(b) Cannot be modelled.

Fig. 4: Evaluation of shape fitting on YCB objects.

B. Parametric Grasp Synthesis

The parametric grasp planner used in this work is inspired by [8]. The grasp configurations are generated perpendicular to a discrete set of object surface normals that cover most of the graspable directions. The planner uses basic parameters like linear and angular grasp distribution around the symmetry axis. In this work, the parametric planner is extended with additional considerations. The hand closure signature, as explained in [18], is computed for the hand and provided as an input to the planner to find grasps that will result in a force closure grasp. The closure signature can be basically interpreted as a fixed transformation that the hand should maintain with respect to the surface normal of the object at the grasping position. This is important especially for anthropomorphic hands, since they only have one opposing thumb finger and the orientation of the hand greatly affects the success of the obtained grasps. Fig. 5d shows the reference frame used for the Touch Bionics i-limb Revolution hand used in TORO, which reflects the main direction provided by the closure signature. The maximum opening of the hand, i.e. the distance between the opposing fingers when they are stretched out, is another property used by the planner. This helps to avoid grasp generation along certain dimensions of the shape that are too big for the hand. A parameter called safe edge distance is also introduced, to constrain the fingers at a certain minimum fixed distance from the edges of the object, in order to avoid too flimsy grasps that might lead to losing the object. The grasp planner also has the possibility to define a fixed number of grasps

to be generated uniformly, or a fixed distance between each one of the grasps. Fig. 5 shows the grasps generated for three primitive shapes using this method. On average, it takes about 10 milliseconds to generate 200 grasps on a normal desktop PC. For complex objects with a very coarse shape estimation, the generated grasps could fail due to no force-closure. Also, the closing of the hand is currently a synchronized movement of all fingers and it is the same for all objects. Integrating a grasp success/failure feedback to trigger a recovery strategy is considered as a future work.

C. Grasp Selection

The reachability and capability maps proposed in [13] are used for the grasp selection process. The reachability map can be queried to find if a particular transformation of the end effector is reachable or not. The capability map additionally provides an index (usually from 0 to 1) reflecting the dexterity of the end effector at that position (i.e. the number of orientations reachable in that voxel). Both maps are generated offline for a given manipulator, and the data can be efficiently retrieved online; approximately 2000 queries can be made every millisecond on a normal desktop PC.

First, the grasps that are generated for the approximated object are filtered by removing the colliding and not reachable (retrieved from the reachability map) ones. Each of the remaining grasps receives a grasp quality, depending on two factors. The first one is derived from the capability of the manipulator at the grasping position, as retrieved from the capability map. The second one is the change in Cartesian distance and orientation of the grasping configuration from the current configuration. The two factors are combined with an appropriate weighting to get the final grasp quality. The weight is empirically selected depending on the particularities of the manipulator's overall workspace. For instance, TORO has only a 6 DoF arm, and the 7th DoF is coming from the hip. With the 7 DoF, TORO has a high dexterous region at the height of the chest, and a medium dexterous region at the height of the hip (which is also the table height approximately). Therefore, the capability index is given a relative high weight of 70%, while the Cartesian distance measure receives a weight of 30%.

IV. OPTIMAL BASE REPOSITIONING

Many a times, humanoid robots have to relocate their base in order to successfully perform a manipulation task at different parts of the environment. For a bipedal robot, we consider the base as the $SE(2)$ position of the hip (height of the hip on the z-axis is constant and rotations are allowed only around the z-axis). The reachability and capability maps are used for computing the optimal base position. The decision to relocate the robot is triggered by having no reachable grasps, or reachable but not executable grasps, i.e. a valid joint trajectory is not possible from the current configuration. The algorithm computes the optimal base position such that it is favorable for a set of possible grasps, rather than for a single predefined grasp. Moreover, it also considers the uncertainty in reaching the base position.

In the offline phase, semantic information of the environment, in this case the table, is saved. The stored info describes a finite set of base positions around the table where interaction with the object is desirable. Online, the set of parametric grasps generated for the object is filtered in order to remove those colliding with the environment. Now, for each potential grasp, a grasping region is defined by a cube around the target position. The size of the cube is decided based on the uncertainty in reaching a goal base position precisely. This can be empirically estimated by commanding various goals and finding the positioning error of the robot. Now, a base dexterity score is computed and used to prioritize the best position of the robot around the table. Using the capability and reachability maps, all the voxels within the grasping region are studied in order to find the average capability index c_i and the number of reachable voxels r_i with the required grasping orientation, respectively. The base dexterity score for a specific grasp is then given by

$$d_i = r_i * c_i. \tag{1}$$

The overall dexterity for a given base position is computed by adding the individual dexterity for each of the n grasps,

$$d = \sum_{i=0}^{i=n} d_i. \tag{2}$$

Algorithm 1 Finding Optimal Base Position

Given: Capability map CM, reachability map RM, cube size c, grasp frames $G = \{G_1, \ldots, G_n\}$, and a set of possible hip frames $H = \{H_1, \ldots, H_m\}$
Output: Optimal hip frame H_{opt} for successful grasping
1: **for each** hip frame H_j in H **do**
2:     **for each** grasp frame G_i in G **do**
3:         $c_i = queryAverageCapability(CM, c, H_j, G_i)$
4:         $r_i = queryReachability(RM, c, H_j, G_i)$
5:         Base dexterity for grasp G_i is $d_i = r_i * c_i$
6:     **end for**
7:     Overall base dexterity $d = \sum_{i=0}^{i=n} d_i$
8:     Save the overall base dexterity to a set D
9: **end for**
10: Sort the set D in descending order
11: return H_{opt} with the best overall base dexterity

Algorithm 1 provides an overview of the algorithm used to compute the best base position. By computing the base position in this way, there is no dependency on a particular grasp. Since a grasping region is considered for every grasp, if the robot could reposition itself within the allowed error threshold, then there is a higher probability of finding more executable grasps. The computation time is proportional to the number of voxels to be queried in the map, which in turn is proportional to the number of grasps, number of base positions considered, and the size of the grasping region. For the experiment discussed in Sec. VI-A, it took 100 milliseconds to compute the optimal base position.

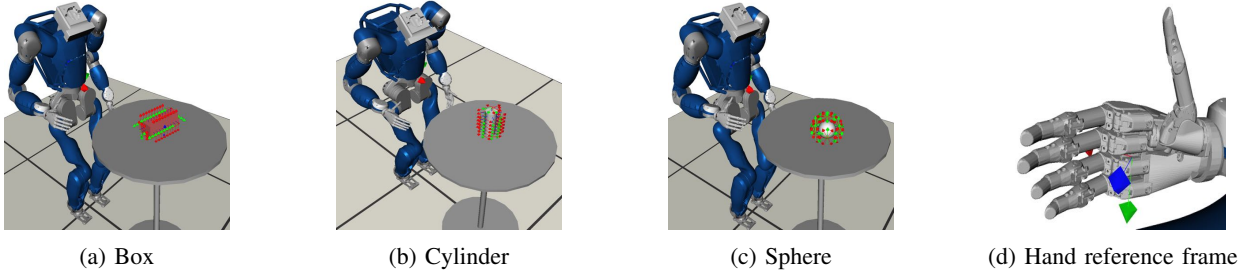(a) Box   (b) Cylinder   (c) Sphere   (d) Hand reference frame

Fig. 5: Grasps generated for the Touch Bionics i-limb (right) hand for different objects.

V. COMPLIANT BALANCING CONTROLLER

A. Dynamic Model

In humanoid robotics, it is common to use a floating-base description of the robot in order to deal with contact transitions or relocations. In most cases, the hip or trunk is selected as the base link, as they represent the central parts of the robot body. But as the center of mass (CoM) is crucial for balancing, from a controller perspective it can also be selected as the base of the robot. For this reason, a frame $\mathcal{C}$ located at the CoM but with the same orientation as the hip is introduced in [19]. Let $\boldsymbol{v}_{\mathrm{c}} \in \mathbb{R}^6$ denote the translational and rotational velocity of the CoM frame $\mathcal{C}$. Based on the n joint angles $\boldsymbol{q} \in \mathbb{R}^n$, the dynamics of the humanoid robot is given by

$$\boldsymbol{M} \begin{pmatrix} \dot{\boldsymbol{v}}_{\mathrm{c}} \\ \ddot{\boldsymbol{q}} \end{pmatrix} + \boldsymbol{C} \begin{pmatrix} \boldsymbol{v}_{\mathrm{c}} \\ \dot{\boldsymbol{q}} \end{pmatrix} + \begin{pmatrix} m\boldsymbol{g}_0 \\ \mathbf{0} \end{pmatrix} + \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\tau} \end{pmatrix} + \boldsymbol{\tau}_{\mathrm{ext}}. \quad (3)$$

where $\boldsymbol{M} \in \mathbb{R}^{(6+n) \times (6+n)}$ and $\boldsymbol{C} \in \mathbb{R}^{(6+n) \times (6+n)}$ denote the inertia and Coriolis/centrifugal matrix, respectively. The gravitational vector is given by $\begin{pmatrix} m\boldsymbol{g}_0^T & \mathbf{0} \end{pmatrix}^T$, where $\boldsymbol{g}_0 \in \mathbb{R}^6$ is the gravitational acceleration and m the total mass of the robot. The joint torques are given by $\boldsymbol{\tau} \in \mathbb{R}^n$, while the influence of external disturbances is represented by the vector of generalized forces $\boldsymbol{\tau}_{\mathrm{ext}} \in \mathbb{R}^{6+n}$.

Stacking the Cartesian velocities $\boldsymbol{v}_i \in \mathbb{R}^6$ (translation and rotation) of the Ψ end-effectors into the vector $\boldsymbol{v} \in \mathbb{R}^{6\Psi}$ leads to

$$\boldsymbol{v} = \begin{bmatrix} \boldsymbol{Ad} & \boldsymbol{J} \end{bmatrix} \begin{pmatrix} \boldsymbol{v}_{\mathrm{c}} \\ \dot{\boldsymbol{q}} \end{pmatrix} \quad (4)$$

with $\boldsymbol{Ad} \in \mathbb{R}^{6\Psi \times 6}$ and $\boldsymbol{J} \in \mathbb{R}^{6\Psi \times n}$ gathering the associated Adjoint $\boldsymbol{Ad}_i \in \mathbb{R}^{6 \times 6}$ and Jacobian matrices $\boldsymbol{J}_i \in \mathbb{R}^{6 \times n}$. Let us divide the Ψ end-effectors into two groups. The ψ end-effectors that are in contact with the environment and used to support the robot will be referred to as "balancing end-effectors". The remaining end-effectors will be called "interaction end-effectors", because they can be used to perform the desired grasping or manipulation task. The velocities of the end-effectors in Cartesian space are then

$$\boldsymbol{v} = \begin{pmatrix} \boldsymbol{v}_{\mathrm{bal}} \\ \boldsymbol{v}_{\mathrm{int}} \end{pmatrix} = \begin{bmatrix} \boldsymbol{Ad}_{\mathrm{bal}} & \boldsymbol{J}_{\mathrm{bal}} \\ \boldsymbol{Ad}_{\mathrm{int}} & \boldsymbol{J}_{\mathrm{int}} \end{bmatrix} \begin{pmatrix} \boldsymbol{v}_{\mathrm{c}} \\ \dot{\boldsymbol{q}} \end{pmatrix} \quad (5)$$

where $\boldsymbol{Ad}$ is decomposed into $\boldsymbol{Ad}_{\mathrm{bal}} \in \mathbb{R}^{6\psi \times 6}$, $\boldsymbol{Ad}_{\mathrm{int}} \in \mathbb{R}^{6(\Psi - \psi) \times 6}$, and $\boldsymbol{J}$ is decomposed into $\boldsymbol{J}_{\mathrm{bal}} \in \mathbb{R}^{\psi \times n}$, $\boldsymbol{J}_{\mathrm{int}} \in \mathbb{R}^{6(\Psi - \psi) \times n}$. Note that $\boldsymbol{v}_{\mathrm{bal}} = \mathbf{0}$ because the balancing end-effectors are in contact with the ground floor.

B. Balancing Controller

This section gives a short summary of the passivity-based balancing controller that the authors presented in [19]. The controller stabilizes the CoM by a Cartesian compliance, which applies a wrench $\boldsymbol{F}_{\mathrm{c}} \in \mathbb{R}^6$ at the CoM frame $\mathcal{C}$. Each one of the interaction end-effectors is stabilized by another Cartesian compliance, with the resulting wrenches stacked into $\boldsymbol{F}_{\mathrm{int}} \in \mathbb{R}^{6(\Psi - \psi)}$. In order to support the robot, the control algorithm computes a suitable set of balancing wrenches $\boldsymbol{F}_{\mathrm{bal}} \in \mathbb{R}^{6\psi}$ by solving the following quadratic optimization problem

$$\boldsymbol{F}_{\mathrm{bal}}^{\mathrm{opt}} = \underset{\boldsymbol{F}_{\mathrm{bal}}}{\arg\min} \left(\boldsymbol{F}_{\mathrm{bal}} - \boldsymbol{F}_{\mathrm{bal}}^{\mathrm{def}} \right)^T \boldsymbol{Q} \left(\boldsymbol{F}_{\mathrm{bal}} - \boldsymbol{F}_{\mathrm{bal}}^{\mathrm{def}} \right) \quad (6)$$

with respect to

$$\boldsymbol{Ad}_{\mathrm{bal}}^T \boldsymbol{F}_{\mathrm{bal}}^{\mathrm{opt}} + \boldsymbol{Ad}_{\mathrm{int}}^T \boldsymbol{F}_{\mathrm{int}} = m\boldsymbol{g}_0 - \boldsymbol{F}_{\mathrm{c}} \quad (7)$$

and

$$\boldsymbol{A}_{\mathrm{bal}} \boldsymbol{F}_{\mathrm{bal}}^{\mathrm{opt}} \leq \boldsymbol{b}_{\mathrm{bal}}. \quad (8)$$

Here, $\boldsymbol{Q}$ denotes a positive definite weighting matrix, and $\boldsymbol{F}_{\mathrm{bal}}^{\mathrm{def}}$ a default wrench distribution. The equality constraint (7) ensures that all end-effector wrenches ($\boldsymbol{F}_{\mathrm{bal}}$ and $\boldsymbol{F}_{\mathrm{int}}$) sum up to the wrench of the CoM compliance plus gravity compensation. The inequality constraint (8) represents the contact model, which accounts for unilaterality, friction and the Center of Pressure (CoP) of each contact in order to prevent the balancing end-effectors from sliding, tilting, or lifting off.

After computing a suitable wrench distribution $\boldsymbol{F}_{\mathrm{bal}}^{\mathrm{opt}}$, the end-effector wrenches $\boldsymbol{F} = \left((\boldsymbol{F}_{\mathrm{bal}}^{\mathrm{opt}})^T \boldsymbol{F}_{\mathrm{int}}^T \right)^T$ are mapped to joint space via

$$\boldsymbol{\tau} = -\boldsymbol{J}^T \boldsymbol{F} - \boldsymbol{N} \boldsymbol{\tau}_{\mathrm{null}}. \quad (9)$$

The torque resulting from the Cartesian end-effector wrenches is superimposed with a torque $\boldsymbol{\tau}_{\mathrm{null}} \in \mathbb{R}^n$, which is included to stabilize the configuration of the robot in nullspace in order to deal with redundant robots. The associated nullspace projector is given by $\boldsymbol{N} \in \mathbb{R}^{n \times n}$.

C. Planner-Controller Interface

In this work, the right manipulator (right arm and hip) is considered as the major interaction end-effector. The left arm and the legs are not considered for the grasping application presented here, but could be included for more complex

manipulation tasks. To reach a desired grasp in the Cartesian space, a valid joint configuration is found using TRAC-IK [20] and fed to a standard bidirectional RRT joint space planner to find a collision-free path. In order to reduce the complexity of the planner, a stationary hip is assumed to be the base of the robot, which allows to do the planning only in the joint space. The trajectory in joint space is post-processed via forward kinematics in order to provide the Cartesian goals for the CoM and the hand compliances in the controller, as explained below.

The controller is connected to the planner via an interface that contains two different types of control objectives. The first one is given by a desired trajectory in Cartesian space for the CoM frame $\mathcal{C}$ and for both hands, which are used to parametrize the Cartesian compliances $\boldsymbol{F}_\mathrm{c}$ and $\boldsymbol{F}_\mathrm{int}$. The second goal is a desired trajectory in joint space, which is fed to the nullspace controller incorporated into (9). The purpose of the balancing controller is to stabilize the desired trajectory while providing a compliant behavior in case of external disturbances. If the controller succeeds to stabilize the desired trajectory provided by the planner, the CoM and the hands will follow exactly their desired path, which causes the hip to maintain a stationary Cartesian pose as assumed in the planning process. If there is an external perturbation, the controller will try to recover the prescribed trajectories. As shown in Sec. VI-B, the controller is robust enough to achieve this task successfully. It should also be noted that the assumption of a stationary hip during the planning phase can theoretically lead to a desired trajectory in which the CoM leaves the support polygon, thus causing the robot to loose balance. To avoid this, the validity of the CoM position is checked before commanding the execution, which is stopped in case of an invalid CoM. But according to the empirical experience of the authors, the CoM always stays well inside the support polygon for the targeted grasps within the reachable workspace of a single interaction end-effector (see Sec. VI-B). This might fail for complex manipulation tasks involving multiple interaction end-effectors, and therefore a suitable joint or Cartesian space planner considering the floating base would have to be used.

VI. Experimental evaluation

In order to validate the approach proposed in this paper, experiments were conducted on the torque-controlled humanoid robot TORO [5]. The robot has 6 DoF in each leg and arm, 1 DoF in the hip, and 2 DoF in the neck. All joints except the neck are based on the KUKA-DLR lightweight robot technology, which allow the robot to be controlled in both position and torque control mode. The robot is also equipped with two i-limb Revolution prosthetic hands from Touch Bionics. This anthropomorphic hand has five under-actuated fingers with 1 active DoF for flexion/extension. The thumb also has an additional active rotational DoF. For the experiments below, the right arm of the robot is used. Since the arm only has 6 DoF, which limits its reachability, the hip is also considered as part of the active manipulation chain.

(a) At 15 degrees

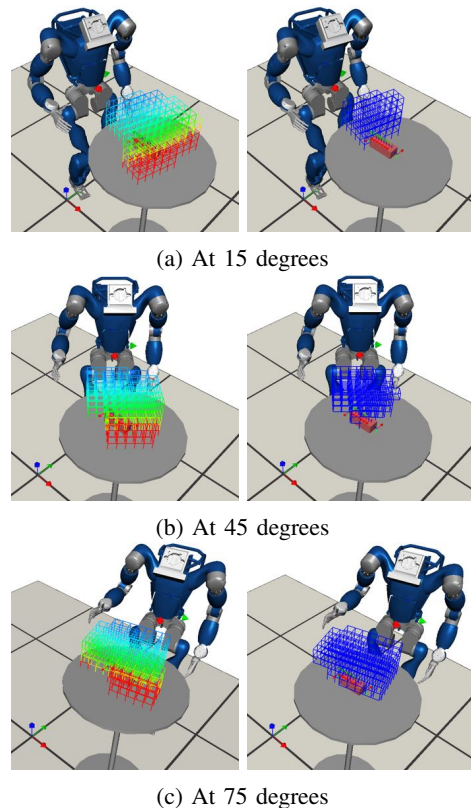(b) At 45 degrees

(c) At 75 degrees

Fig. 6: Capability (left column, colors are denoted in the HSV scale from blue being the highest dexterity to red being the lowest dexterity) and reachability (right column, only the voxels reachable with the required grasping orientation are highlighted in blue) of the right arm plus hip for the grasping regions of all non-colliding grasps at different base positions during Experiment I. The different base positions are considered around the circular table at different angles while maintaining the same distance from the table.

The robot is equipped with an Asus Xtion RGB-D sensor on the head, which is used for perceiving the environment.

A. Experiment I: Base Repositioning

This experiment shows the robot grasping a box using base repositioning. The robot starts from a initial location and walks to the table identified using an AprilTag. After reaching the table, the robot finds the object successfully and generates a set of parametric grasps. Due to the kinematic limitations of the manipulator, there were no reachable grasps possible. The robot then computes the best base position for the current object location using the algorithm described in Sec. IV. For all the grasp regions corresponding to the set of all non-colliding grasps initially generated, Fig. 6 shows the capability and reachability at different base positions during the planning phase. In this experiment, 7 non-colliding grasps on top, 1 in front and 1 in the back of the object were considered. The reachability and capability maps were generated offline with a voxel resolution of $5cm$. The cube size (representing the uncertainty in the base position
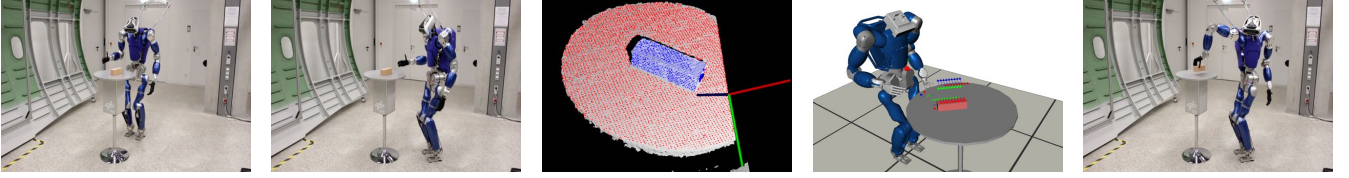
Fig. 7: Sequence of actions during Experiment I. From left to right: (i) The robot successfully walks and places itself in front of a circular table, (ii) Since there are no executable grasps, the robot moves to the optimal base position, (iii) The shape and pose of the object are detected, (iv) Reachable and non colliding grasps are generated, (v) The grasp with the best quality and a valid motion plan is executed.

reachability) of the grasping region was estimated as $20cm$. Fig. 6a to Fig. 6c show that the dexterity indicated by the capability index and the number of reachable voxels with the grasping orientation are gradually increasing according to the angular position around the circular table. Note that the highlighted voxels in Fig. 6 correspond to the position of the end of the arm for each potential grasping point, and do not represent the hand position. This is due to the fact that the capability and reachability maps were generated only for the right arm plus hip, without considering the hand.

After finding the best base position, the robot starts walking to it step by step using a simple Cartesian space footstep planner. ORB-SLAM2 [12] is used to provide visual odometry and localize the robot. As expected, the robot approximately reaches the best base position. In order to proceed with the grasping, the object shape and pose are estimated again. The grasps are generated and the best executable grasp is selected. For this grasping pose, the motion plan is generated and fed to the whole-body balancing controller for grasp execution. The different execution steps of the experiment are shown in Fig. 7.

B. Experiment II: Compliant Control

This experiment shows the robot grasping an apple while handling external disturbances and perturbations. The apple was identified as a sphere. In order to avoid collisions with the table, the planner selected a grasp from the top. Since the apple was located on the table within the reach of the robot, relocating the base was not necessary. Besides the grasping aspect, this experiment is mainly used to demonstrate the importance of using a compliant whole-body balancing controller as part of the autonomous grasping pipeline, in order to deal with external disturbances. During the execution of the grasp trajectory, the robot received a push at the hip. Then, the hand of the robot was restrained to simulate a collision with an obstacle. The resulting position and orientation errors of the right hand and the CoM are shown in Fig. 8. The push at the hip created a deviation of $3.6\,\mathrm{cm}$ at the CoM, but the controller successfully manages to keep the disturbance at the CoM away from the right hand. After the push, the hand was held back in its motion for approximately $1\,\mathrm{s}$, which resulted in a deviation of $3.4\,\mathrm{cm}$ of the hand position. This time, the CoM was not affected by the disturbance. After the right hand was released, the robot continued its task while correcting the error, and successfully picked up

the apple from the table. This highlights the benefits of using a compliant balancing controller, such that the robot can be safely operated in a partially unknown or uncertain environment, even in the presence of external disturbances such as pushes or collisions. A deeper discussion on the performance of the controller can be found in [19].
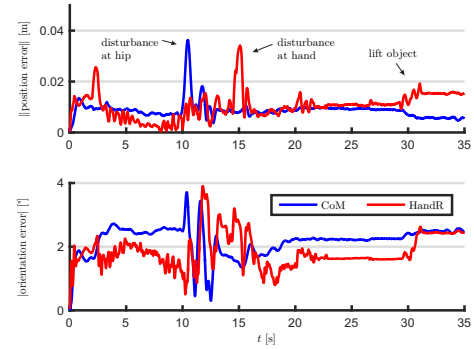

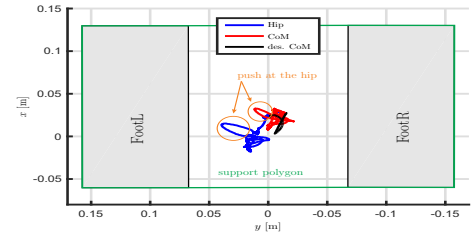Fig. 8: Control error for CoM and right hand during Experiment II.

Fig. 9: Trajectory of the hip and the CoM projected to the ground floor during Experiment II.

Figure 9 shows a projection of the CoM and hip motion onto the ground floor during the execution of the grasp trajectory. It clearly shows that the CoM always stays far inside the support polygon spanned by the two feet. As a consequence, selecting a stationary hip during the planning phase, which leads to a moving CoM, is a legitimate assumption as postulated in Sec. V-C. Figure 9 also visualizes the push that the robot received at the hip, causing a temporary deviation of the hip and the CoM from the planned trajectory. Note that except for the push, the hip almost remains stationary at its initial position, as it is intended by the planning algorithm (see Sec. V-C). The different execution steps of the experiment are shown in Fig. 10.
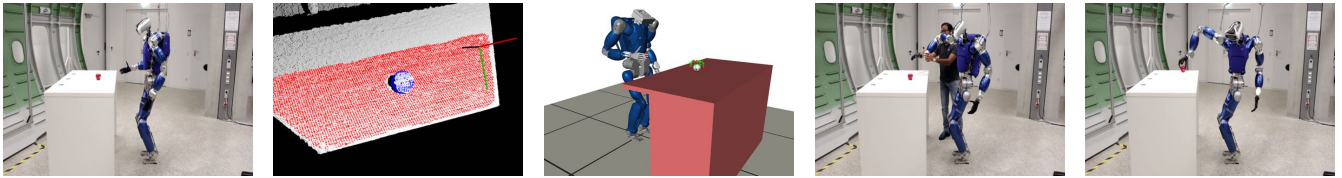
Fig. 10: Sequence of actions during Experiment II. From left to right: (i) Starting configuration in front of a rectangular table, (ii) The shape and pose of the object are detected, (iii) Reachable and non colliding grasps are generated, (iv) The compliant balancing controller successfully handles the external disturbances, (v) The grasp with the best quality and a valid motion plan is executed.

VII. Conclusion

This paper presented an overall approach to create autonomous grasping behaviors in bipedal humanoid robots. Parametric grasping based on primitive shapes was used for online generation of candidate grasps. A reachability analysis was used to compute the best grasp poses. More importantly, the reachability analysis was also used to compute the best pose for relocating the base of the robot such that the grasping probability can be increased for a given scenario. The base repositioning considers multiple grasp poses and uncertainty in reaching the calculated optimal base position precisely. A compliant whole-body controller allows a robust execution of the intended task while handling the external disturbances, as demonstrated in the experiments in the paper. To validate the robustness and flexibility of the approach, similar experiments were conducted to grasp different objects, some of which are also shown in the attached video.

Current base repositioning works on SE(2), but the extension of the method to SE(3), to profit from the full potentiality of the humanoid robot and the whole-body controller, is an ongoing work. The extension of the framework for dual arm manipulation and the implementation of grasp failure recovery strategies are considered for future work.

Acknowledgment

References

[1] K. Harada and M. A. Roa, "Manipulation and task execution by humanoids," in *Humanoid Robotics: A Reference*, A. Goswami and P. Vadakkepat, Eds. Springer, 2016, pp. 1–23.

[2] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Robot placement based on reachability inversion," in *Proc. IEEE Int. Conf. Robotics and Automation - ICRA*, 2013, pp. 1970–1975.

[3] F. Burget and M. Bennewitz, "Stance selection for humanoid grasping tasks by inverse reachability maps," in *Proc. IEEE Int. Conf. Robotics and Automation - ICRA*, 2015, pp. 5669–5674.

[4] Y. Yang, W. Merkt, H. Ferrolho, V. Ivan, and S. Vijayakumar, "Efficient humanoid motion planning on uneven terrain using paired forward-inverse dynamic reachability maps," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2279–2286, 2017.

[5] J. Englsberger, A. Werner, C. Ott, B. Henze, M. A. Roa, G. Garofalo, R. Burger, A. Beyer, O. Eiberger, K. Schmid, and A. Albu-Schäffer, "Overview of the torque-controlled humanoid robot TORO," in *Proc. IEEE-RAS Int. Conf. on Humanoid Robots*, 2014, pp. 916–923.

[6] A. Makhal and A. K. Goins, "Reuleaux: Robot base placement by reachability analysis," in *Proc. IEEE Int. Conf. Robotic Computing - IRC*, 2018, pp. 137–142.

[7] A. Hertle and B. Nebel, "Identifying good poses when doing your household chores: Creation and exploitation of inverse surface reachability maps," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS*, 2017, pp. 6053–6058.

[8] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic grasp planning using shape primitives," in *Proc. IEEE Int. Conf. Robotics and Automation - ICRA*, vol. 2, 2003, pp. 1824–1829.

[9] M. Nieuwenhuisen, J. Stückler, A. Berner, R. Klein, and S. Behnke, "Shape-primitive based object recognition and grasping," in *Proc. German Conf. on Robotics - ROBOTIK*. VDE, 2012, pp. 1–5.

[10] K. Huebner, K. Welke, M. Przybylski, N. Vahrenkamp, T. Asfour, D. Kragic, and R. Dillmann, "Grasping known objects with humanoid robots: A box-based approach," in *Proc. IEEE Int. Conf. Advanced Robotics - ICAR*, 2009, pp. 1–6.

[11] J. Bohg and D. Kragic, "Grasping familiar objects using shape context," in *Proc. IEEE Int. Conf. Advanced Robotics - ICAR*, 2009, pp. 1–6.

[12] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[13] O. Porges, T. Stouraitis, C. Borst, and M. A. Roa, "Reachability and capability analysis for manipulation tasks," in *ROBOT2013: First Iberian Robotics Conference*. Springer, 2014, pp. 703–718.

[14] Z.-C. Marton, D. Pangercic, N. Blodow, J. Kleinehellefort, and M. Beetz, "General 3D Modelling of Novel Objects from a Single View," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS*, 2010, pp. 3700–3705.

[15] S. Büttner, Z.-C. Márton, and K. Hertkorn, "Automatic scene parsing for generic object descriptions using shape primitives," *Robotics and Autonomous Systems*, vol. 76, pp. 93–112, 2016.

[16] M. Schuster, S. Brunner, K. Bussmann, S. Büttner, A. Dömel, M. Hellerer, H. Lehner, P. Lehner, O. Porges, J. Reill, S. Riedel, M. Vayugundla, B. Vodermayer, T. Bodenmüller, C. Brand, W. Friedl, I. Grixa, H. Hirschmüller, M. Kassecker, Z. Marton, C. Nissler, R. Felix, M. Suppa, and A. Wedler, "Towards autonomous planetary exploration: The Lightweight Rover Unit (LRU), its success in the Spacebotcamp challenge, and beyond," *J. Intelligent & Robotic Systems*, pp. 1–34, 2017.

[17] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The YCB object and model set: Towards common benchmarks for manipulation research," in *Proc. IEEE Int. Conf. Advanced Robotics - ICAR*, 2015, pp. 510–517.

[18] M. Pozzi, G. Salvietti, J. Bimbo, M. Malvezzi, and D. Prattichizzo, "The closure signature: a functional approach to model underactuated compliant robotic hands," *IEEE Robotics and Automation Letters*, pp. 2206–2213, 2018.

[19] B. Henze, M. A. Roa, and C. Ott, "Passivity-based whole-body balancing for torque-controlled humanoid robots in multi-contact scenarios," *Int. J. Robotics Research*, vol. 35, no. 12, pp. 1522–1543, 2016.

[20] P. Beeson and B. Ames, "TRAC-IK: An open-source library for improved solving of generic inverse kinematics," in *Proc. IEEE-RAS Int. Conf. on Humanoid Robots*, 2015, pp. 928–935.