Planning Heavy Lifts for Humanoid Robots*

Michael Grey[1] Sungmoon Joo[2] Matt Zucker[3]

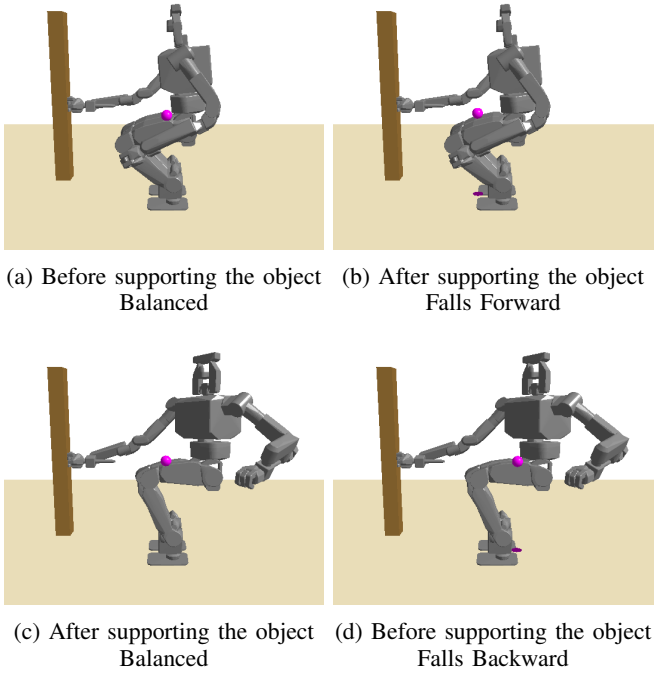
***Abstract*— Lifting heavy objects poses a unique challenge for humanoid robots and more broadly, for any robot which is responsible for maintaining its own balance. Configurations that are balanced without supporting a heavy object's weight might not be balanced while the object's weight is being supported, and vice versa. In this paper, we present a series of planning techniques which resolve these issues without relying on real time control methods or extensive force/torque sensing. We introduce the novel concept of the Virtual Task Dimension (VTD) for motion planners, which can handle the transition between balancing constraints. We describe the implementation of these techniques and offer suggestions for obtaining fast and reliable solutions. We also demonstrate the algorithms running on a DRC-Hubo humanoid robot.**

I. Introduction

Interest in using humanoid robots to perform intense physical labor has been growing in recent years, especially with the DARPA Robotics Challenge (DRC) pushing the state of the art in robot autonomy. One of the challenges in the DRC Trials required the robot to clear wooden blocks out of a tightly constricted area as seen in Figure 2.

Previous work has addressed the challenge of heavy lifting and carrying for humanoid robots. In [1] the robot's wrist and ankle force sensor information is used to generate whole body motions which can track a desired gait pattern and Zero Moment Point (ZMP) trajectory while carrying a heavy object of unknown weight. This idea is built upon in [2] where the robot exploits singular configurations to minimize joint torque. A contact force based approach to compliant heavy lifting is given by [3]. However, none of these methods address the challenge of avoiding obstacles throughout the heavy lifting task; they assume an open, collision-free environment. Additionally, all of these methods rely heavily on sensor feedback, placing a burden on the user to tune control gains so that they exhibit the desired response properties.

In the interest of quickly generating robot motions for heavy lifts which avoid obstacles and require minimal real time control and sensing for success, we build on the task constrained motion planning [4] and Constrained Bi-directional Rapidly exploring Random Tree (CBiRRT) [5] framework. CBiRRT is a probabilistically complete motion planning algorithm which is able to rapidly generate motion plans in high dimensional space while respecting task constraints. Since humanoid robots are high degree of freedom systems with a wide variety of constraints, CBiRRT is well suited for planning in this domain.



(a) Before supporting the object Balanced (b) After supporting the object Falls Forward

(c) After supporting the object Balanced (d) Before supporting the object Falls Backward

Fig. 1: The robot needs to change its configuration before it can support the weight of the object. The purple sphere is the CoM of the configuration, and the disk on the ground is its projection.

However, traditional use of CBiRRT cannot plan the transition from the configuration seen in Figure 1a to the configuration in Figure 1c. It can only handle the heavy lifting problem if there exists a picking configuration that is valid both with and without the target object's weight being supported (see Figure 4a). If the target object is heavy or the robot's support polygon is too small, such a configuration might be difficult or impossible to find (see Figure 4b). Therefore, while performing a heavy lift, the planner must be able to account for the transition from supporting none of the target object's weight to supporting all of its weight, and this can be thought of as a change in the mass model of the balancing constraint. In order to plan this transition, we introduce the concept of the Virtual Task Dimension (VTD). The VTD is a dimension appended to the configuration space which represents some additional piece of context for the planner. In the case of heavy lifting, the

*This work was supported by DARPA award D13AP00047

[1]Michael Grey is a doctoral student in the School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332 mxgrey@gatech.edu

[2]Sungmoon Joo is a research scientist in the School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332 sungmoon.joo@cc.gatech.edu

[3]Matt Zucker is an Assistant Professor in the Department of Engineering, Swarthmore College, Swarthmore, PA 19081 mzucker1@swarthmore.edu

VTD is used to represent the fraction of the target object's weight which the robot is supporting. Including the VTD in the configuration space enables a planner to explore this dimension and therefore plan out the mass model transition. The theory behind the VTD is covered in greater detail in subsection II-C.

In section II we discuss all the relevant constraints for lifting heavy objects, how to use those constraints to find start and goal configurations, and how the Virtual Task Dimension fits into this framework. In section III we describe our implementation and some practical considerations for quickly generating reliable plans. In section IV we demonstrate the algorithm working in two distinct scenarios on the DRC-Hubo humanoid robot. Finally, section V finishes with a discussion of potential future work.

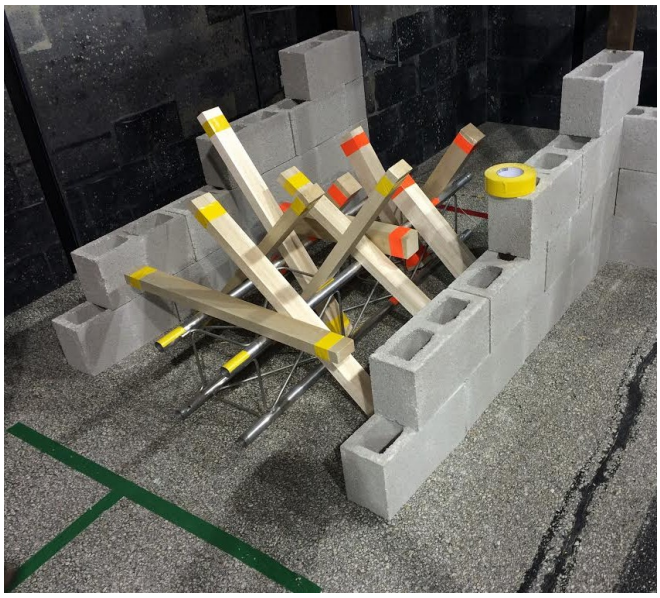


Fig. 2: Debris Removal Task at the DARPA Robotics Challenge Trials

II. Theory

Generating a motion plan involves solving for start/goal configurations and then searching for a path between them. Start and goal configurations must satisfy the constraints of the initial and final objectives respectively (for example, reaching for an item or placing an item on a table). For an entire path to be valid, each point along the path must satisfy the feasibility constraints, such as being balanced and collision-free. In this section, we discuss how to formulate the constraints needed to plan heavy lifts, how to use those constraints to find start/goal configurations, and finally how to use the Virtual Task Dimension to plan a heavy lift.

A. Constraint Functions

There are four types of constraints which are crucial to the heavy-lifting problem: Balancing constraints, end-effector pose constraints, collision constraints, and joint torque constraints. To resolve these constraints, we use recursive hierarchical nullspace projection [6].

1) Balance Constraints: A humanoid robot must keep its center of mass above its support polygon in order to maintain quasistatic balance. In some scenarios, the balance constraint could be maintained by rejecting configurations which violate it. However, in the case of heavy lifting, it is unreasonable to simply hope that balanced configurations will be sampled, so a projection operation is needed. We use the center of mass Jacobian method to drive the robot's center of mass into its support polygon.

2) End-effector Constraints: In order to handle end-effector constraints, we employ Task Space Regions [7]. Task Space Regions provide a convenient and generalized way of describing and solving end effector constraints, including articulated constraints. A critical example of how TSRs are used for planning heavy lifts is for finding kinematically feasible grasp configurations. We assume that any given target object has a set of viable grasp points or grasp regions and then we represent those points or regions as a set of TSRs.

Additionally, we must constrain both feet to remain stationary during a plan. Any relative motion between the feet while both are being used for support would produce dangerous internal forces inside the robot which could result in mechanical or electrical damage. In this paper, we do not consider the possibility of taking a step during a plan; we leave the challenges of taking steps and selecting foot placement for future work.

3) Collision Constraints: In wide-open environments, collision constraints may be handled by simply rejecting configurations which exhibit collisions. Alternatively, using convex collision geometries allows configurations to be projected out of collision using gradient descent like in [8] and [9]. Escaping collisions using gradient descent can be useful in cluttered environments where the passages between obstacles are thin.

4) Torque Constraints: A fourth constraint which may be important to consider for heavy lifting (depending on the strength of the robot) is torque constraints. Since physical motors have torque limits, any configuration whose joint torques exceed those limits must be considered invalid. The probability of randomly sampling a configuration which violates torque limits is low enough that simply rejecting invalid configurations (instead of trying to project them) should be suitable. The Jacobian Transpose can be used to compute the static joint torques for a given configuration.

B. Finding Start/Goal Configurations

Before CBiRRT can generate a path, it is necessary to determine start/goal configurations. These configurations must satisfy all of the feasibility constraints (i.e. they must be balanced and collision-free) in addition to achieving an objective. The objective can also be represented as a constraint, ideally as a Task Space Region, which describes where the robot should grab or where it should deliver its payload. Because of the high dimensionality of humanoid robot systems and the nonlinearity of the constraint functions, analytical

solutions for satisfying all of these constraints simultaneously do not generally exist.

Therefore, in order to solve all the simultaneous constraints, we used stochastic gradient descent. In wide-open, easy environments it might be possible to simply use ordinary gradient descent, but when dealing with cluttered or challenging environments, it is common to get caught in local minima.

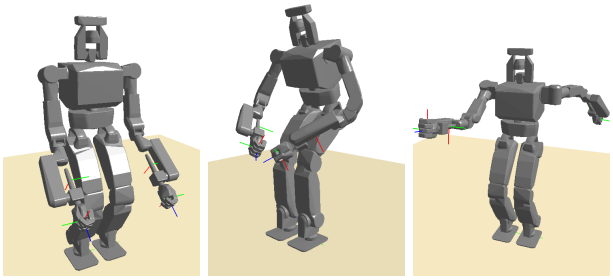


Fig. 3: Examples of useful seeding configurations.

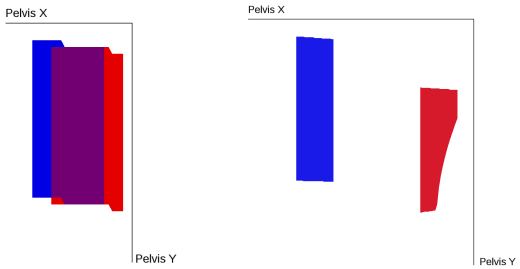
Rather than beginning gradient descent attempts from random configurations, we start them from seeding configurations which are known to be far from singularities or local minima. Samples of the seeding configurations which were used in this project can be seen in Figure 3.

In order to plan a heavy lift, it is necessary to find two lifting configurations: one which satisfies all constraints *before* supporting the object's weight, and one which satisfies all constraints *while* supporting the object's weight.

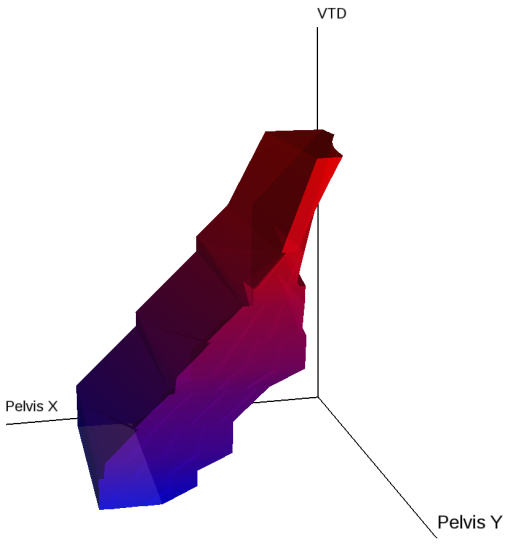
C. Planning with a Virtual Task Dimension

Once start and goal configurations are determined, we use CBiRRT [5] to find paths between them. In some scenarios, it may be possible to find picking configurations where the robot is balanced both with and without the weight of the target object. This is often the case if the target object is light-weight in comparison to the robot, or if the robot has a very large support polygon. Figure 4a shows an example where the valid configurations before and after have an overlap (the purple region). To plan between configurations whose constraint functions do not have an overlap, we introduce the concept of the Virtual Task Dimension (VTD). Figure 4c shows an approximate projection of valid configurations across the VTD for heavy lifting.

The VTD is a continuous dimension which is appended to the robot's configuration space. A configuration which supports none of the object's weight has a VTD value of 0.0 while a configuration which supports all of the object's weight has a VTD value of 1.0. As the CBiRRT planner explores the robot's configuration space, it will also explore the VTD. As it does, the constraints which are applied to the robot's configurations will change according to the VTD value; specifically, as the VTD value increases, the balance constraint will need to support more of the target object's weight. The planner can explore the VTD freely, moving arbitrarily forward and backward through the dimension, however the "Connect" operation in CBiRRT makes this



(a) Light-weight object (b) Heavy object

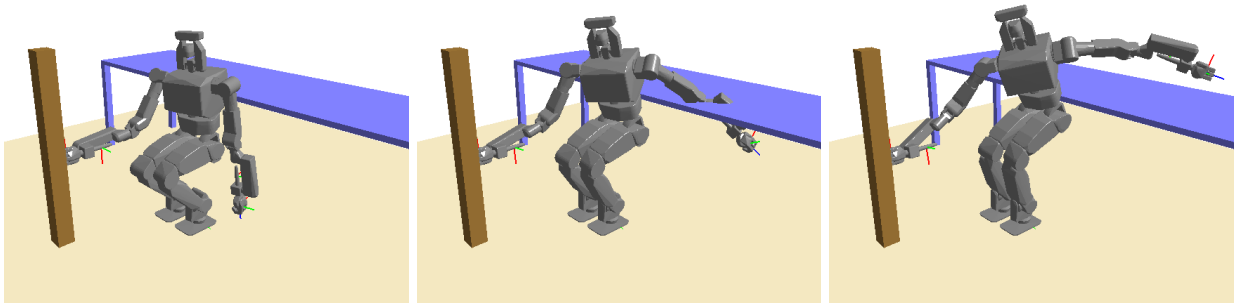


(c) Projection of valid Pelvis X/Y translations along with VTD for the same heavy object as Figure 4b.

Fig. 4: These are projections of two components (Pelvis X and Y) from the 22 dimensional configuration space during a lift. Pelvis X and Y are the X/Y translations of the robot's root Pelvis link. Blue regions are valid while the robot supports none of the object's weight; red regions are valid while the robot supports all of the object's weight. In Figure 4a the purple region represents an overlap of the blue and red regions, indicating that there do exist lifting configurations that are balanced both with and without the object's weight supported. Figure 4b has no such intersection, therefore there is no way to plan a valid path from the blue to the red without introducing the VTD. Figure 4c shows the space of valid configurations for Figure 4b once the VTD is introduced; the planner is able to explore this space freely in order to find a valid path from a point in the blue region to a point in the red region.

behavior rare, and path shortening during post-processing will tend to reduce the VTD traversal into a straight line if possible.

The VTD is needed as a dimension in the planner in order to find feasible paths between the start and goal configurations. Gradient descent or naive interpolation may fail to produce valid connections in the case of obstacles or local minima, as seen in Figure 5b. If such obstacles do not exist, then using CBiRRT will be as fast as a naive interpolation, because CBiRRT employs bi-directional connection attempts between the configurations, which simply interpolates between the points. This means that CBiRRT will be as efficient as a naive approach in the scenarios when a naive approach would succeed while still being



(a) Valid configuration without supporting the object's weight (b) Invalid interpolation between start and goal configurations (c) Valid configuration while supporting the object's weight

Fig. 5: Example of why planning the transition between start and goal configurations is necessary.

probabilistically complete and therefore capable of tackling more challenging scenarios than what a naive approach could handle.

III. Implementation

Whole body planning is generally difficult due to its high dimensionality and tight constraints. In this section, we will discuss the particular way we approached the whole body planning problem, highlighting various techniques which help to make it tractable. Some of these techniques are based on our particular use of DRC-Hubo, but may also apply to other humanoid robots.

A. Floating Base Method

Each leg on DRC-Hubo has 6 degrees of freedom, and as mentioned in subsubsection II-A.2, we constrain both feet to remain stationary. This means that 12 dimensions of the joint space are fully constrained. If these joints were included in the configuration space that is used by the planner, an inordinate amount of time would be spent projecting the legs' joint values onto their constraint manifold. Instead, we leave the legs' joint angles out of the configuration space used for planning, and simply use the six degrees of freedom (three translational and three rotational) of the robot's pelvis (root) link. Imagine that the robot's legs are removed and the pelvis could float around and rotate itself freely.

Analytical inverse kinematics is used to ensure that the robot's floating-base poses are feasible. If no valid leg joint configuration exists for a given pelvis pose, the analytical IK is guaranteed to report it. Moreover, even though a 6-DoF analytical IK will usually produce 8 unique joint configurations, the legs on the DRC-Hubo can only ever have up to one valid joint configuration once the legs' joint limits are taken into account. This means that there is no redundancy to account for while generating a plan, and so the planner is *still probabilistically complete* even without explicitly considering these degrees of freedom. We implicitly plan the joint angles of the legs by planning the six degrees of freedom of the pelvis and verifying that there exists a valid set of leg configurations for any given pelvis pose.

B. Task Decomposition

We decompose the lifting task into four phases: Approach, Lift, Deliver, and Release.

1) Approach: The Approach phase can be thought of as an ordinary picking task. The entire task must take place at a VTD value of 0. This can be done either by removing the VTD from the configuration space or by setting both the minimum and maximum values of the VTD domain to 0.

2) Lift: The Lift phase is where the robot transitions from supporting none of the object's weight to supporting all of its weight. In other words, this is where the plan will travel from a VTD value of 0.0 to a VTD value of 1.0. The start configuration of the Lift phase must be the goal configuration of the Approach phase, and the goal configuration of the Lift phase will be used as the start configuration of the Deliver phase.

3) Delivery: The Delivery phase can be thought of as an ordinary placement task. In the Delivery phase, the VTD value must always be 1.0.

4) Release: The Release phase is used when the final objective is for the robot to let go of the target object. The Release phase is exactly like the Lift phase, except that it traverses from a VTD value of 1.0 to a value of 0.0.

C. Trajectory Generation

Using CBiRRT generates a path through configuration space, but execution on a robot requires a trajectory which specifies joint values as a function of time. For this we used a time-optimal trajectory generation algorithm by Kunz [10]. This algorithm accepts as input a path as well as maximum speeds and accelerations for each dimension in the configuration space. For the maximum speeds and accelerations, we chose small values to ensure that the robot always maintains quasistatic behavior.

IV. Experiments

To demonstrate the theories presented, we present two distinct heavy lifting experiments using a DRC-Hubo. In the first, the robot needs to reach forward and lift a heavy wooden block while avoiding a nearby obstacle. In the

second experiment, the robot needs to use both hands to lift a heavy metal truss onto a small platform.

No force-torque sensors were used in either experiment. The robot was running through the trajectory using only joint encoder feedback with relatively stiff PD gains. All of the robot's balancing was due to the planner. In general, we consider force-torque sensor feedback to be a useful tool for maintaining balance, but we refrained from using it for these experiments in order to demonstrate the utility of our kinematic-only planning approach.

A. DRC-Hubo

The robot platform used for these experiments is the DRC-Hubo. The DRC-Hubo is a humanoid robot made by Rainbow, Inc. and KAIST in South Korea. It was designed as an upgrade to its predecessor the Hubo2+ with the goal of competing in the DARPA Robotics Challenge. The robot has a mass of roughly 55 kg and stands at 147 cm. Each leg has 6 degrees of freedom and each arm has 7. There is one additional degree of freedom which allows the torso to rotate about the vertical axis.

B. Block Lifting Problem

In the first task, DRC-Hubo must lift a 4.04 kg (7.34% of the robot's mass) wooden block which was located approximately 65 cm away from the robot. Additionally, there is a table behind the robot which acts as an obstacle. The table's presence makes it impossible for the robot to shift its pelvis back far enough to compensate for the weight of the block. The planner automatically decides to instead use the left arm to counter-balance the weight of the block, as can be seen in Figure 6a.

C. Truss Lifting Problem

The second task had DRC-Hubo lift a 4.01 kg metal truss onto a small metal platform. The dimensions of the truss are 31 cm x 31 cm x 152 cm, and it can be seen in Figure 6d. This task demonstrates the ability of the planner to handle dual-arm manipulation constraints.

D. Timing Results

Runtimes for random sampling methods like CBiRRT are variable, so timing data were collected for 100 runs of each task and are laid out in Table I. The dual-arm manipulation constraint for the Truss Lifting Problem has a much lower dimensional constraint manifold than the single-arm manipulation of the Block Lifting Problem, so it generally took longer to solve.

	Block		Truss	
	Finding Goals	Planning	Finding Goals	Planning
Average	32.24	6.16	60.92	54.94
Min	6.14	0.63	34.95	6.21
Max	133.23	48.62	151.70	358.55
Std Dev	24.91	8.90	23.23	60.49

TABLE I: Timing data for the two experiments. All units are in seconds.

V. Conclusions and Future Work

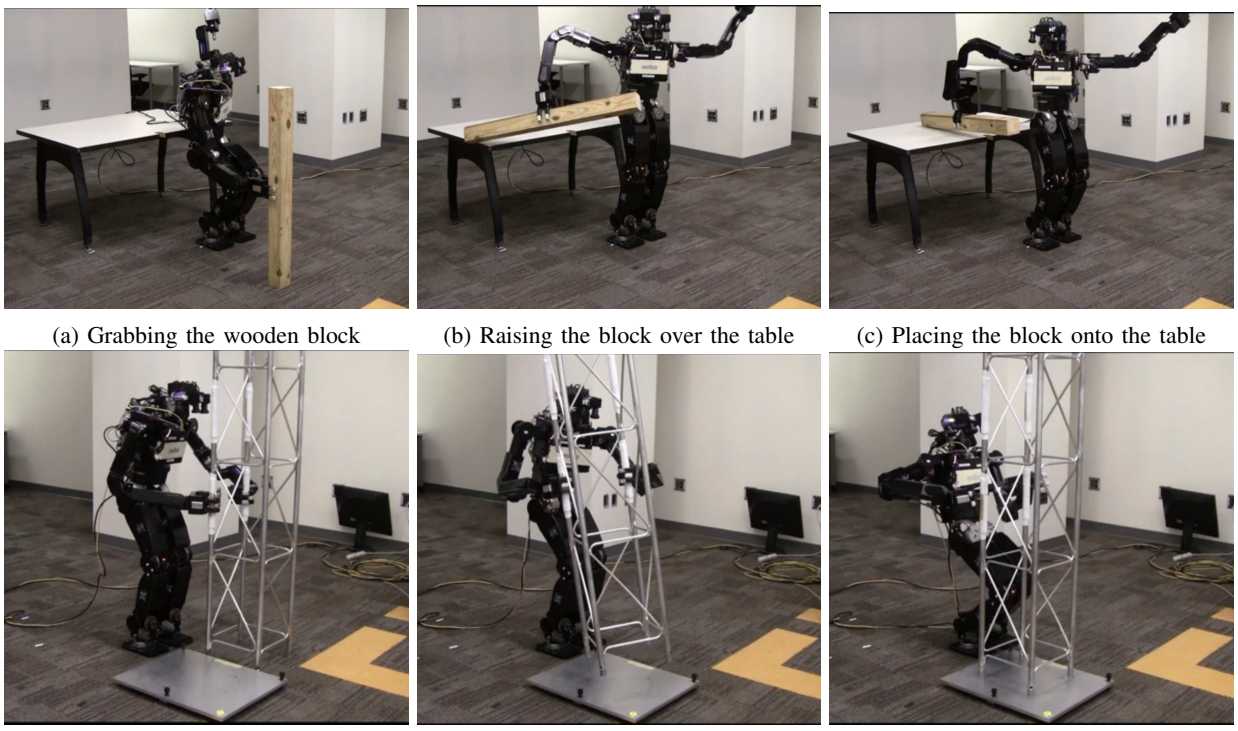
This work introduces the concept of the Virtual Task Dimension for use in heavy lifting. We describe a comprehensive and probabilistically complete method for autonomously planning heavy lifts. This method was demonstrated in two distinct trials on a physical DRC-Hubo. Prior methods for achieving heavy lifts with a humanoid robot focused heavily on real time feedback control and neglected to account for obstacles in the environment. The method we present generates a feasible path ahead of time, which gives far greater assurance that the robot's actions will be successful. We do not consider this method to be a replacement for the existing real-time control methods; instead we see it as a way to provide stronger guarantees that those methods will perform effectively by giving them a feasible baseline trajectory.

In future work, we intend to address the question of autonomously deciding upon foot placement for the lift, and perhaps being able to take steps during the plan. A hierarchical planning methodology [11] might suit these challenges well, coupled with task-driven support polygon reshaping [12].

One major weakness of our planning method is that it assumes prior knowledge of the target object's mass. A single plan may be valid for a range of object masses, depending on the size of the robot's support polygon, but for heavy objects, a small percentage of error in the estimate could easily make a plan invalid. To overcome this issue, we will investigate methods for generating contingency plans for a broad range of potential object masses and finding connections between the plans to switch between them while the robot is interacting with the object.

We also believe that the Virtual Task Dimension concept has far broader applications than the heavy lifting problem. The VTD effectively enables a CBiRRT framework to handle constraints which change as a function of time or some other contextual parameter which is not normally considered part of the robot's configuration space. An example which is strongly related to heavy lifting would be dragging; the VTD would allow the planner to reason about the trade-off in joint torques between lifting (which will reduce the friction force) and pushing/pulling. Another related example would be negative VTD, which would enable the planner to reason about leaning some of its weight onto an object in the environment while still maintaining balance.

In future work, we will investigate alternative methods of generating dynamically stable trajectories from the planned path which do not rely on moving slowly. For example, AutoBalancer [13] was used in [14] to generate dynamically balanced trajectories using RRTs.



(a) Grabbing the wooden block (b) Raising the block over the table (c) Placing the block onto the table

(d) Grabbing the metal truss (e) Raising the truss over the platform (f) Placing the truss onto the platform

Fig. 6: DRC-Hubo performing heavy lift tasks

ACKNOWLEDGMENTS

This work was funded by the DARPA Young Faculty Award given to Mike Stilman, who tragically passed away before this project started. Nevertheless, we know he would have been proud of our achievements and excited to see where this work leads.

We would also like to thank Andrea Thomaz and Aaron Bobick whose moral, administrative, and technical support have been invaluable in these difficult times.

REFERENCES

[1] K. Harada, S. Kajita, H. Saito, M. Morisawa, F. Kanehiro, K. Fujiwara, K. Kaneko, and H. Hirukawa, "A humanoid robot carrying a heavy object," in *Proc. IEEE Int'l. Conf. on Robotics and Automation*. IEEE, 2005, pp. 1712–1717.

[2] H. Arisumi, S. Miossec, J.-R. Chardonnet, and K. Yokoi, "Dynamic lifting by whole body motion of humanoid robots," in *Proc. IEEE/RSJ Int'l. Conf. on Intelligent Robots and Systems*. IEEE, 2008, pp. 668–675.

[3] B. J. Stephens and C. G. Atkeson, "Dynamic balance force control for compliant humanoid robots," in *Proc. IEEE/RSJ Int'l. Conf. on Intelligent Robots and Systems*. IEEE, 2010, pp. 1248–1255.

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

[5] D. Berenson, S. Srinivasa, D. Ferguson, and J. Kuffner, "Manipulation planning on constraint manifolds," in *IEEE International Conference on Robotics and Automation (ICRA '09)*, May 2009, pp. 625–632.

[6] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," in *International Journal of Humanoid Robotics*, 2005, pp. 505–518.

[7] D. Berenson, S. Srinivasa, and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *International Journal of Robotics Research (IJRR)*, vol. 30, no. 12, pp. 1435 – 1460, October 2011.

[8] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *ICRA'09. IEEE International Conference on Robotics and Automation, 2009*. IEEE, 2009, pp. 489–494.

[9] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[10] T. Kunz and M. Stilman, "Time-optimal trajectory generation for path following with bounded acceleration and velocity," 2012, p. 209.

[11] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical planning in the now," in *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[12] E. Yoshida, O. Kanoun, C. Esteves, and J.-P. Laumond, "Task-driven support polygon reshaping for humanoids," in *6th IEEE-RAS International Conference on Humanoid Robots, 2006*, Dec 2006, pp. 208–213.

[13] S. Kagami, F. Kanehiro, Y. Tamiya, M. Inaba, and H. Inoue, "Autobalancer: An online dynamic balance compensation scheme for humanoid robots," in *Fourth Int. Workshop on Algorithmic Foundations on Robotics*, 2000.

[14] J. J. Kuffner Jr, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue, "Dynamically-stable motion planning for humanoid robots," *Autonomous Robots*, vol. 12, no. 1, pp. 105–118, 2002.