Randomized Multi-Modal Motion Planning for a Humanoid Robot Manipulation Task

Article *in* The International Journal of Robotics Research · May 2011

DOI: 10.1177/0278364910386985 · Source: DBLP

2 authors, including:

Victor Ng-Thow-Hing
Magic Leap
50 PUBLICATIONS **1,198** CITATIONS

Multi-Modal Motion Planning for a Humanoid Robot Manipulation Task

Kris Hauser[1], Victor Ng-Thow-Hing[2], and Hector Gonzalez-Baños[2†]

[1] Computer Science Department, Stanford University, Stanford, CA USA
khauser@cs.stanford.edu
[2] Honda Research Institute, Mountain View, CA USA
vng@hri.com, hhg@4espi.com
[†] Currently affiliated with Electronic Scripting Products, Inc., 260 Sheridan Ave, Suite B25, Palo Alto, CA USA

Abstract. This paper presents a motion planner that enables a humanoid robot to push an object on a flat surface. The robot's motion is divided into distinct walking, reaching, and pushing modes. A *discrete* change of mode can be achieved with a *continuous* single-mode motion that satisfies mode-specific constraints (e.g. dynamics, kinematic limits, avoid obstacles). Existing techniques can plan well in single modes, but choosing the right mode transitions is difficult. Search-based methods are vastly inefficient due to over-exploration of similar modes. Our new method, *Random-MMP*, randomly samples mode transitions to distribute a sparse number of modes across configuration space. Results are presented in simulation and on the Honda ASIMO robot.

1 Introduction

Pushing is a potentially useful form of manipulation for humanoid robots when grasping is impossible. But pushing is not as simple as walking to the object and moving the arm; advance planning is crucial. Even simple tasks, like reorienting the object in place, may require a large number of pushes. Between pushes, the robot may need to switch hands or walk to a new location, choosing carefully among alternatives so that each push respects kinematic constraints and avoids collision. Furthermore, many tasks cannot

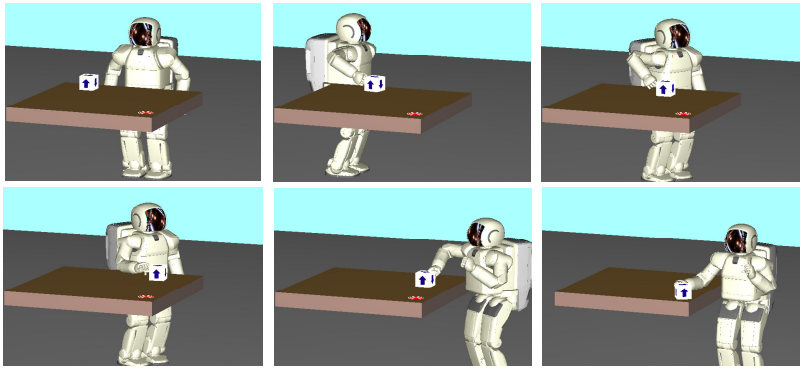

Fig. 1. To cross a table, an object must be pushed along the table's edges

be solved by greedily pushing the object toward its target. For example, an object cannot be pushed directly across a large table (Fig.1). Once the object out of reach it cannot be pushed further, and since pushing is nonprehensile, the object cannot be recovered.

The current technology in ASIMO's control system requires dividing the robot's motion into distinct walk, reach, and push *modes*. While ASIMO walks, the swaying of its body prohibits accurate hand positioning. Therefore, the robot is required to stand still while reaching for and pushing the object. Secondly, to predict the object's motion when pushed, we restrict ourselves to use *stable pushes* [7]. This imposes additional constraints on the hand and object motion during a push.

Given these constraints, a motion planner must produce a *discrete* sequence of modes, as well a *continuous* motion through them. This *multi-modal planning* problem occurs in several areas of robotics. In manipulation planning, motion alternates between transfer and transit (object grasped/not grasped) modes [1, 8, 9]. In legged locomotion, each set of environment contacts defines a mode [2, 4]. Modes also occur in reconfigurable robots [3] and as sets of subassemblies in assembly planning [10]. The most general existing multi-modal planning approach first appeared in manipulation planning as a "manipulation graph" [1], and can be described as *mode-before-motion* search. It constructs a graph of modes by selecting an existing mode, and transitioning to neighboring modes with single-mode motions. However, in pushing and other problems, some modes have a continuous set of neighbors (e.g. to start pushing, any points on the surface of the hand and object can meet). A fixed discretization makes search intractable, even for simple push tasks.

The problem is not that pushing itself is hard, but that search samples modes much too densely. Inspired by probabilistic motion planners, the novel *Random-MMP* approach samples mode transitions at random, according to a strategy designed to distribute modes sparsely across configuration space. A simple blind strategy samples transitions (roughly) uniformly at random. Though this is easy to implement and performs reasonably well, it can be improved with prior knowledge of the push task. After precomputing tables of push utility – the expected distance the object can be pushed – we bias the sampling of contact points to yield high-utility pushes. We additionally focus on "bottlenecks" by picking good pushes before choosing where to walk. The combined strategy plans for difficult problems in minutes on a PC. We demonstrate results in simulation and experiments on the real robot.

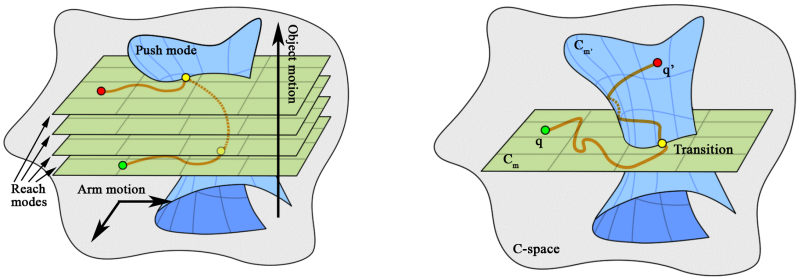

Fig. 2. (a) Abstract depiction of reach and push modes, with arm motion horizontal, object motion vertical. Each object configuration yields a new reach mode. (b) Paths from q to q' pass through a transition configuration in $F_m \cap F_{m'}$

2 Problem Specification

We plan for ASIMO to push an object across a horizontal table. We move one arm at a time for convenience. We assume the object moves quasi-statically (slides without toppling and comes to rest immediately), and can be pushed without affecting the robot's balance. The planner is given a per-fect geometric model of the robot, the object, and all obstacles. Other physical parameters are specified, e.g. the object's mass and the hand-object friction coefficient. Given a desired translation and/or rotation for the object, it computes a path for the robot to follow, and an expected path for the object.

2.1 Configuration Space

A configuration q combines a robot configuration q_{robot} and an object configuration q_{obj}. ASIMO's walking subsystem allows fully controllable motion in the plane, so leg joint angles can be ignored. Thus, q_{robot} consists of a planar transformation $(x_{robot}, y_{robot}, \theta_{robot})$, five joint angles for each arm, and a degree of freedom for each hand ranging from open to closed. Since the object slides on the table, q_{obj} is a planar transformation $(x_{obj}, y_{obj}, \theta_{\text{obj}})$. In all, the configuration space C is 18 dimensional.

The robot is not permitted to collide with itself or obstacles, and may only touch the object with its hands. It must obey kinematic limits. The object may not collide with obstacles or fall off the table. We also require that the object be visible from the robot's cameras while pushing to avoid some unnatural motions (e.g. behind-the-back pushes).

2.2 Modes and submanifolds

The robot's motion is divided into five mode classes: walking, reach left, reach right, push right, and push left. Each mode has its own motion dynamics and constraints, specified as follows. In *walk* modes, only the base of the robot $(x_{robot}, y_{robot}, \theta_{robot})$ moves. The arms must be raised to a "home configuration" that avoids colliding with the table while walking. In

reach modes, only a single arm and its hand may move. In *push* modes, the hand is in contact with the object. The object moves in response to the arm motion according to push dynamics, and reciprocally, the dynamics impose constraints on arm motions (Sect. 2.4). Additionally, the object must lie in the robot's field of view.

Each mode constrains motion to a submanifold of lower dimension than C. Let C_m denote the submanifold corresponding to mode m, and F_m denote the set of configurations in C_m that satisfy all feasibility constraints of m. An important semantic note is that a mode m refers to both the *mode class* as well as *all other parameters* necessary to fully describe the motion constraints. For example, there are an infinite number of reach modes, each one with a distinct object position (Fig. 2.a). We represent modes (nonuniquely) by an integer describing the mode class and a representative configuration.

2.3 Adjacencies and transitions

We say modes m and m' are *adjacent* if a transition is permitted between them. For a path to transition from m to m', some configuration q along the way must satisfy the constraints of both modes. An important consequence is that the intersection of F_m and $F_{m'}$ must be nonempty (Fig. 2.b). We call $q \in F_m \cap F_{m'}$ a *transition configuration.*

The following mode transitions are permitted (Fig. 3). *Walk-to-reach*, r*each-to-reach*, and *push-to-reach* are allowed from any feasible configuration. Either the left or right arm may be chosen. *Reach-to-walk* is allowed if the arms are returned to the home configuration. *Reach-to-push* is allowed when the hand of the moving arm contacts the object.

2.4 Push Dynamics

We restrict the planner to use pushes that, under basic assumptions, rotate the object predictably. These *stable pushes* must be applied with at least two simultaneous collinear contacts, such as flat areas on the robot's hand. Given known center of friction, surface friction, and contact points, one can calculate simple conditions on the stable centers of rotation (CORs) [7]. Rotating the hand in the plane about a stable COR c will predictably rotate the object about c. Pure translations are represented by a COR at infinity.

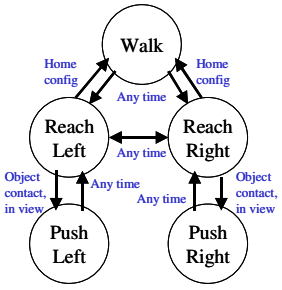

Fig. 3. Mode transition diagram

3 Multi-Modal Planning

3.1 Single-mode planning

Single-mode motions can be planned quickly with standard techniques. On average, each plan takes a small but not negligible amount of time

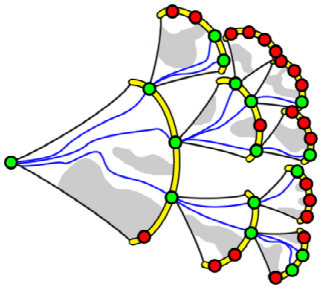

Fig. 4. Diagram of search with a fixed discretization. Each "fan" is a mode's configuration space. Yellow regions are continuous sets of potential transitions. Feasible transition configurations are green, infeasible are red. Blue lines are single-mode paths

(typically between 10 and 100 ms). Walk modes are 3D, and motions can be planned with a variety of methods. Reach modes are 6D, requiring the use of probabilistic roadmap (PRM) methods. PRMs build a roadmap of randomly sampled, feasible configurations, connecting them with straight-line paths. They plan quickly when the space has favorable *visibility* properties [5], which are almost always satisfied in reach modes. However, PRMs cannot determine that no path exists, so the planner declares failure after a specified time limit.

Push motions are produced as follows. Let p_{hand} be a contact on the hand touching a point p_{obj} on the object, with normals n_{hand} and n_{obj}. First, sample a stable COR c. Rotate the object about c for some distance. Maintain hand contact during this rotation using a numerical inverse kinematics (IK) solver to position p_{hand} at p_{obj} and orient n_{hand} to $-n_{obj}$. If the motion is feasible, repeat the process to push the object further.

3.2 Existing multi-modal approaches

Some multi-modal problems can be solved with standard PRMs simply by allowing a mode-change action. This action succeeds only at transition configurations in regions $F_m \cap F_{m'}$. But most interesting multi-modal systems contain transitions $F_m \cap F_{m'}$ with lower dimension than F_m (or $F_{m'}$, or both). In particular, a reach-to-push transition requires that a flat part of the hand touch the object. The set of all such configurations has zero measure in the 6D reach submanifold, so a randomly sampled arm configuration has zero probability of transitioning to a push. This necessitates *mode-before-motion* approaches, which explicitly consider mode transitions as targets for single-mode planning.

The most general mode-before-motion approach is based on classical search, and can work well if good mode-based heuristics are developed. The method builds a search tree T where nodes are configuration/mode pairs. At each step, the method picks an unexpanded node (q,m) from T according to a heuristic, and expands it as follows. For each adjacent mode m', plan a single-mode path y in m, starting at q and ending at a transition q' in $F_m \cap F_{m'}$. If successful, add the edge $(q,m) \rightarrow (q',m')$ to T, annotated

with *y*. Once a goal is reached, the motion follows the single-mode motions along the edges of the solution path.

Search is directly applicable if each mode has a finite number of adjacencies [2]. For systems with continuously varying modes, the system must be discretized because each mode has an uncountable number of adjacencies [1, 4, 8] (Fig. 4). Choosing a discretization requires trading off between speed and completeness.

A notable alternative to discretizing continuous sets of modes is based on a roadmap of the set of configurations that transition between *any* two modes [1, 9]. Unfortunately, a controllability condition has restricted this method so far to prehensile manipulation.

3.3 Drawbacks of search

Using search for push planning requires discretizing walk positions (walk-to-reach transitions), contacts (reach-to-push transitions), and pushes (push-to-reach transitions). But even the sparsest discretization makes search intractable. Consider pushing a box. One should allow either hand to touch each side of the box, so there must be $k \geq 2$ hand contacts and $m \geq 4$ box contacts. One should allow at least a straight push and CW and CCW rotations, so there must be $n \geq 3$ pushes. Finally adding p walk positions (say, $p \geq 4$), a search tree of d pushes expands $O((kmnp)^d)$ modes. In terms of pushes, the branching factor is no less than 96. Since each expansion takes 10-100 ms, even expanding to a depth of two pushes is too costly. Furthermore, it appears difficult to develop good heuristics, because they must reason about the feasibility of future transitions and single-step paths.

4 Random-MMP

If pushing were truly intractable, search might be our only option. But search covers the configuration space much more densely than is needed, e.g. pushing left with the right hand from position x is similar to pushing left with the left hand from position y. In continuous spaces, PRMs use randomness to overcome similar discretization issues. Their performance depends on the *visibility* properties of the space [5], and if visibility is good, a roadmap of a small number of configurations sampled at random is sufficient to capture the connectivity of the space. This inspires the development of Random-MMP. We conjecture that pushing and other multi-modal systems exhibit good "visibility", although more work is needed to define such a term in the multi-modal case.

Like mode-before-motion search, Random-MMP maintains a tree T and extends it with a single-mode transition. But each extension picks a node from T at random with probability Φ, and expands to a single adjacent mode sampled at random with probability Ψ. But how to select Φ and Ψ?

Some intuition can be gained by examining how tree-growing PRMs try to sample in low-density areas. Two early planners have been empirically shown to have good performance over a wide range of problems, and have inspired dozens of variations existing in the literature. The EST planner expands from a node with probability inversely proportional to the number of nearby nodes [5]. The RRT planner tries to distribute configurations uniformly across the space by picking a random point in space, and expanding the closest node toward that point [6]. We use a simple RRT-like implementation, expanding the tree as follows:

Random-MMP

1. Sample a random configuration q_{rand}.
2. Pick a node (q,m) that minimizes a distance metric $d(q,q_{rand})$. We define d to be the distance of the object configurations, ignoring the robot entirely. Like RRT, step 2 implicitly defines Φ proportional to the size of the Voronoi cell of q.
3. The tree is expanded from (q,m) to new mode(s) using an expansion strategy *Expand*, which implicitly defines Ψ.

We will compare four variants of the expansion strategy *Expand*.

- *Blind:* Expands to an adjacent mode m' chosen at random.
- *Reach/Utility-Informed:* Same, but samples contacts, for reach-to-push transitions, according to expected reachability/utility.
- *Push-centered:* Expands a sequence of modes that executes a high-utility push that moves q_{obj} toward q_{rand}.

5. Expansion Strategies

Blind sampling is important to consider, because it can be used for most multi-modal problems simply by implementing a transition sampler. Reach/utility-informed sampling improves contact selection by assigning probability proportional to reachability/utility, much like importance sampling. These weights are precomputed on grids in the workspaces of contacts on the hand. Push-centered expansion makes a further improvement by selecting body positions that execute high-utility pushes.

5.1 Blind Expansion

Given a configuration q at mode m, blind expansion samples a transition configuration q' at an adjacent mode m', and if q' is feasible, plans a single-mode path y to connect q and q' as usual. We first choose an adjacent mode class, then sample q' to achieve that type of transition (together defining m' as remarked in Sect. 2.2). We sample q' as follows:

- *Walk-to-reach*: Sample a body position between a minimum and maximum distance from the object and a body orientation such that the object lies within the robot's field of view.
- *Reach-to-walk*: Move the arm to the home configuration.
- *Reach-to-reach*: Use any configuration.
- *Reach-to-push*: Let S_{hand} and S_{obj} be the surfaces of the hand and the object. We predefine a number of contact points $C_{cand} \subseteq S_{hand}$ that are candidates for stable pushes. Sample a point p_{hand} from C_{cand} and a point p_{obj} from S_{obj}, with normals n_{hand} and n_{obj}. Starting from a random arm configuration, use numerical IK to simultaneously position p_{hand} at p_{obj} and orient n_{hand} to $-n_{obj}$.
- *Push-to-reach*: Rather than sample and plan separately, we plan the single-mode path y as in Sect. 3.1, and let q' be the endpoint of y.

5.2 Reach/Utility-Informed Sampling

In reach-to-push sampling, only a small portion of S_{obj} is reachable from a given point on the hand. For each p in C_{cand}, we precompute information that helps identify the reachable region R on S_{obj}, and furthermore measures the expected utility of points in R.

When pushing, the normal n at p must be horizontal in world space. We fix a height of pushing h, constraining the vertical coordinate of p. This define a 3D workspace W of points (x,y,θ), where (x,y) are the horizontal coordinates of p and θ is the orientation of n, relative to the robot's frame (Fig. 5.a). We precompute two tables over W as follows.

Reachable stores 1 if the contact is reachable and 0 otherwise (Fig. 5.c). We initialize *Reachable* to 0, and then sample the 5D space of the arm joints in a grid. Starting at each sampled configuration q, we run IK to bring the height of p to h and reorient n to be horizontal. If successful, we check if the arm avoids collision with the body and the point p is in the robot's field of view. If so, we mark *Reachable*$[(x,y,\theta)]$ with 1, where (x,y,θ) are the workspace coordinates of p and $[\cdot]$ denotes grid indexing.

Utility stores the expected distance the contact can be pushed in the absence of obstacles, calculated by Monte Carlo integration through

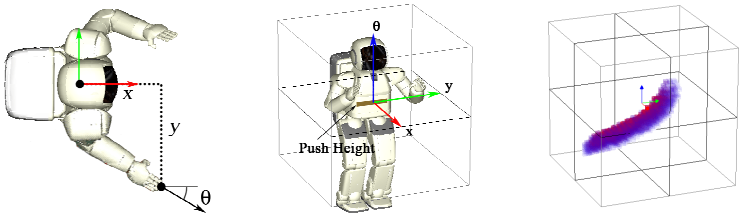

Fig. 5. (a) Workspace coordinates of the right fingers. (b) Reference frame F, with vertical axis indicating rotation angle. (c) Reach/utility table in frame F. Reachable cells are drawn with utility increasing from blue to red

Table 6. Expansion strategy experiments. Bold indicates best in column

	Nodes / push	Time / push (s)	Average push (cm)	Push rate (m/s)	Tgt. seek rate (m/s)
Blind	10.0	0.956	6.7	0.070	0.0302
Reach-informed	6.99	0.353	6.5	0.185	0.0658
Utility-informed	5.99	**0.325**	8.2	0.254	0.111
Push-centered	**5.08**	0.404	**13.3**	**0.329**	**0.257**

Reachable (Fig. 5.c). In W, a push traces out a helix that rotates around some COR. We assume a prior probability Π over stable CORs for a reasonable range of physical parameters of the object. Starting from $w_0=(x,y,\theta)$ we generate a path $w_0, w_1,\ldots, w_K$. For all k, w_{k+1} is computed by rotating w_k a short distance along some COR sampled from Π. The sequence terminates when $Reachable[w_{K+1}]$ becomes 0. After generating N paths, we record the average length in $Utility[(x,y,\theta)]$.

Given robot and object positions, contacts along S_{obj} at height h form a set of curves B in W. Intersecting B with the marked cells of *Reachable* gives the set of reachable object edges R. *Reach-informed sampling* samples contacts uniformly from R. Furthermore, *utility-informed sampling* samples from R with probability proportional to *Utility*.

5.4 Push-centered expansion

During reach-to-push transitions, the maximum push utility depends greatly on the placement of the robot body. A randomly chosen placement may have a hard time reaching the object or pushing it as desired. Push-centered expansion explicitly chooses a good body position to execute a high-utility push. Given a node (q,m), this strategy 1) chooses a robot's body and arm configuration and a high-utility push for a reach-to-push transition q_{push}, 2) plans a whole sequence of modes *backwards* from the reach mode at q_{push} to (q,m), requiring no search, and 3) plans a push path *forward* from q_{push}.

We elaborate on step 1. Let q_{obj} be the object configuration in the randomly sampled configuration q_{rand}. Choose a point p_{obj} on S_{obj} (at height h and normal n_{obj}) and a stable push such that the object will be pushed toward q_{obj}. Next, sample a contact p_{hand} from C_{cand} and a workspace coordinate $(x_{hand},y_{hand},\theta_{hand})$ with probability proportional to *Utility*. Then, compute the body coordinates that transform (x_{hand},y_{hand}) to p_{obj} and rotate θ_{hand} to θ_{obj}, where θ_{obj} is the orientation of $-n_{obj}$. Repeat until the body position is collision free. Fixing the body, sample the arm configuration of q_{push}, using IK to pos-ition p_{hand} to p_{obj} and orient n_{hand} to $-n_{obj}$.

5.5 Experimental Comparison

We measure the performance of an expansion strategy as the distance the object is pushed per unit of planning time. We ran Random-MMP on the setup of Fig. 7 using the blind, reach-informed, utility-informed, and push-centered strategies. The search is initialized with a walk mode, and terminates at a single push (requiring three transitions, from walk to reach to push). The final column measures the distance the object was pushed toward the object position in q_{rand} (if positive). The results, averaged over 1,000 runs, are summarized in Table 6. Though blind expansion is improved with reach- and utility-informed sampling, push-centered expansion is clearly superior.

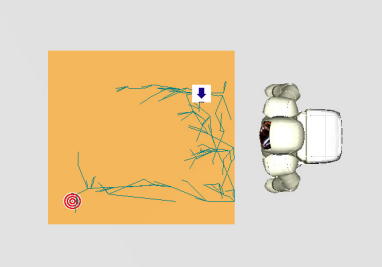

Fig. 7. Search tree for the plan of Fig.1

6 Simulations and Experiments

Figs. 1 and 7 show a generated motion plan in simulation. The goal is to push the object to the opposite table corner. The planner found a trajectory in about one minute on a 2GHz PC. Planning times increase if obstacles introduce difficult bottlenecks, e.g. in Fig. 8. A moved obstacle invalidates the initial path during execution, forcing the robot to a more difficult alternative. The planner produced a new path in three minutes.

We performed tests on the physical robot (Fig. 9) executing the motion without visual or tactile feedback. The object and table have known geometry and are marked with calibration patterns. Only at the start, the object and table are sensed using stereo vision, and the planner is given their transformations relative to the robot. The robot performs several pushes (typically 3 to 5) almost exactly as planned. After a while, drift in the robot position (estimated with dead reckoning) causes pushes to fail. Current hardware can only localize the object and table periodically by walking to a location where all calibration patterns are in view. Future work could modify cameras to provide continuous feedback, introducing the possibility of planning and executing unstable (point) pushes.

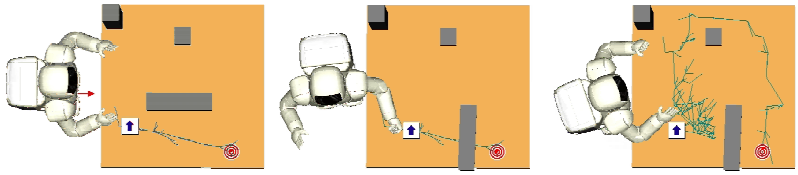

Fig. 8. Replanning in a changing environment

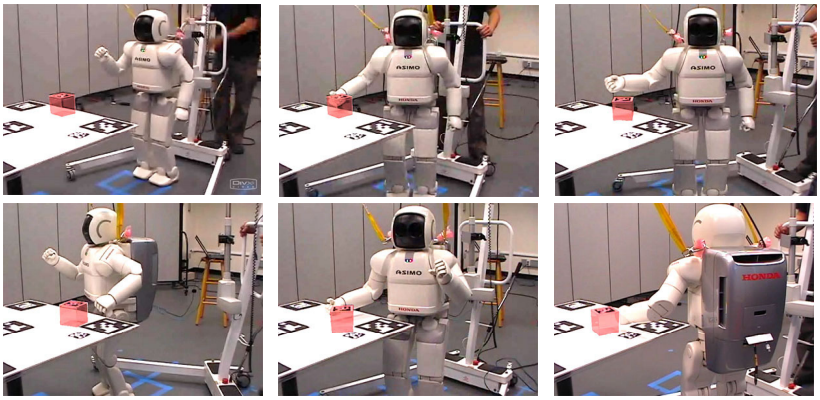

Fig. 9. Asimo pushing a block. Block is highlighted in red for clarity. More videos are available at http://draco.honda-ri.com:8080/Videos/Videos

8 Conclusion

We presented a manipulation planner for a humanoid robot using a novel yet simple multi-modal planning algorithm, Random-MMP. Random-MMP tries to distribute modes sparsely across the configuration space, much like sample-based motion planners. We accelerate planning by biasing transition sampling toward high-utility pushes. Experiments show that even without visual feedback, the motions can be executed reasonably well. Further work might improve planning speed using alternate search strategies, or improve motion quality.

This work also brings up the possibility of unifying multi-modal planning research across application domains. Future work should advance the understanding of the entire spectrum of multi-modal problems, and compare existing approaches across problems. This may enable building efficient, general-purpose multi-modal planners.

Acknowledgements. Jean-Claude Latombe provided helpful comments on the paper. This work was partially supported by NSF grant IIS-0412884.

References

1. Alami R, Laumond JP, Simeon T (1995) Two Manipulation Planning Algorithms. In: Algorithmic Foundations of Robotics, K. Goldberg et al. (eds.). A K Peters, Wellesley (MA), pp 109-125
2. Bretl T (2006) Motion Planning of Multi-limbed Robots Subject to Equilibrium Constraints: The Free-climbing Robot Problem. In: Intl J of Robotics Research, 25(4):317-342
3. Casal A (2001) Reconfiguration Planning for Modular Self-Reconfigurable Robots. PhD Thesis, Aero & Astro Dept, Stanford University, Stanford, CA

4. Hauser K, Bretl T, Latombe J-C (2005) Non-gaited Humanoid Locomotion Planning. In: Proc IEEE Intl Conf on Humanoid Robotics
5. Hsu D, Latombe J-C, Motwani R (1999) Path Planning in Expansive Configuration Spaces. In: Intl J of Computational Geometry, 9(4-5):495-512
6. LaValle SM, Kuffner JJ (1999) Randomized Kinodynamic Planning. In: Proc IEEE Intl Conf on Robotics and Automation, pp 473-479
7. Lynch KM, Mason MT (1996) Stable pushing: Mechanics, controllability, and planning. In: Intl J of Robotics Research, 15(6): 533-556
8. Nieuwenhuisen D, van der Stappen AF, Overmars, MH (2006) An Effective Framework for Path Planning Amidst Movable Obstacles. In: Workshop on Algorithmic Foundations of Robotics
9. Simeon T, Cortes J, Sahbani A, Laumond J-P (2002) A General Manipulation Task Planner. In: Workshop on Algorithmic Foundations of Robotics
10. Wilson R (1992) On Geometric Assembly Planning. PhD Thesis, Stanford University, Stanford, CA