

THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

Scaling Sampling-based Motion Planning to Humanoid Robots

Citation for published version:
Yang, Y, Ivan, V, Merkt, W & Vijayakumar, S 2017, Scaling Sampling-based Motion Planning to Humanoid Robots. in *IEEE International Conference on Robotics and Biomimetics ROBIO 2016.* Institute of Electrical and Electronics Engineers (IEEE), pp. 1448-1454, IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND BIOMIMETICS 2016, Qingdao, China, 3/12/16. https://doi.org/10.1109/ROBIO.2016.7866531

Digital Object Identifier (DOI):
10.1109/ROBIO.2016.7866531

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
IEEE International Conference on Robotics and Biomimetics ROBIO 2016

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

OPEN ACCESS

Download date: 29. Mar. 2024

arXiv:1607.07470v2 [cs.RO] 29 Jul 2016

Scaling Sampling–based Motion Planning to Humanoid Robots

Yiming Yang, Vladimir Ivan, Wolfgang Merkt, Sethu Vijayakumar

***Abstract*— Planning balanced and collision–free motion for humanoid robots is non–trivial, especially when they are operated in complex environments, such as reaching targets behind obstacles or through narrow passages. We propose a method that allows us to apply existing sampling–based algorithms to plan trajectories for humanoids by utilizing a customized state space representation, biased sampling strategies, and a steering function based on a robust inverse kinematics solver. Our approach requires no prior offline computation, thus one can easily transfer the work to new robot platforms. We tested the proposed method solving practical reaching tasks on a 38 degrees–of–freedom humanoid robot, NASA Valkyrie, showing that our method is able to generate valid motion plans that can be executed on advanced full–size humanoid robots. We also present a benchmark between different motion planning algorithms evaluated on a variety of reaching motion problems. This allows us to find suitable algorithms for solving humanoid motion planning problems, and to identify the limitations of these algorithms.**

I. Introduction

Humanoid robots are highly redundant systems that are designed for accomplishing a variety of tasks in environments designed for human. However, humanoids have a large number of degrees–of–freedom (DoF) which makes motion planning extremely challenging. In general, optimization–based algorithms are suitable for searching for optimal solutions even in high dimensional systems [1] [2], but it is non–trivial to generate optimal collision–free trajectories for humanoids using optimization approaches, especially in complex environments. This is mainly due to the highly non–linear map between the robot and the collision environment. This mapping can be modelled in some abstract spaces to provide real–time collision avoidance capabilities on low DoF robotic arms [3] difficult for high DoF humanoids due to the curse of dimensionality and it often causes local minima problems. Additionally, solving locomotion and whole–body manipulation in complex environments as one combined problem requires searching through a large space of possible actions. Instead, it is more effective to first generate robust walking plans to move the robot to a desired standing location, and then generate collision–free motion with stationary feet [4]. Although assuming fixed feet position may be viewed as restrictive, we argue that a large variety of whole body manipulation tasks can still be executed as a series of locomotion and manipulation subtasks. We propose an extension to a family of sampling based motion planning algorithms that will allow us to plan collision–free whole–body motions on floating based systems which require active balancing.

All authors are with School of Informatics, University of Edinburgh (Informatics Forum, 10 Crichton Street, Edinburgh, EH8 9AB, United Kingdom). email: yiming.yang@ed.ac.uk

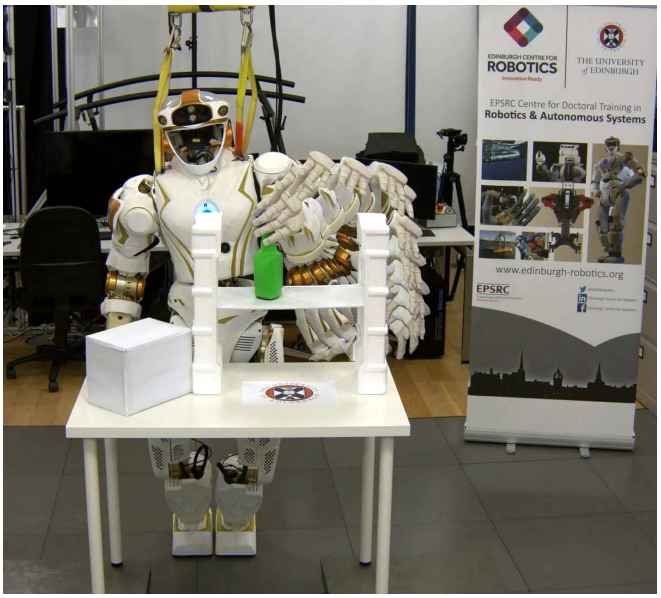


Fig. 1: Collision–free and balanced whole–body motion executed on the 38 DoF NASA Valkyrie robot.

Sampling–based planning (SBP) algorithms, such as RRT [5] and PRM [6], are capable of efficiently generating globally valid collision-free trajectories due to their simplicity. In the past two decades, SBP algorithms have been applied to countless problems with a variety of derivatives, such as RRT-Connect [7], Expansive Space Trees (EST, [8]), RRT*/PRM* [9], Kinematic Planning by Interior-Exterior Cell Exploration (KPIECE, [10]), and many others [11]. However, since the SBP algorithms were originally designed for mobile robots and low DoF robotic arms, using them on high DoF systems requiring active balancing is still challenging. We will call a robot pose statically balanced if the controller can achieve an equilibrium in this state while achieving zero velocity and acceleration (e.g. when the projection of centre of mass lies within the support polygon). The subset of robot configurations with this property forms a low dimensional manifold defined by the balance constraint. In practice, the rejection rate of random samples is prohibitively high without the explicit or implicit knowledge of the manifold. Approaches have been proposed to address this particular problem of using SBP algorithms for humanoid robots. Kuffner at al. [12] use a heavily customized RRT–Connect algorithm to plan whole body motion for humanoids, where they only sample from a pre-calculated

pool of postures for which the robot is in balance. Hauser at al. [13] introduce motion primitives into SBP algorithms where the sampler only samples states around a set of pre–stored motion primitives. A similar approach is used in [14] with centre–of–mass (CoM) movement primitives. These approaches share the common idea of using an offline generated sample set to bootstrap online processes, thus allowing algorithms to bypass the expensive online generation of balanced samples. However, this leads to the problem where one has to store a significant number of samples to densely cover the balance manifold, otherwise the algorithms may fail while valid solutions exist but were not stored during offline processing. Another issue is that the pre–processing is normally platform specific, which makes it difficult and time consuming to transfer the work to other robot platforms.

To this end, instead of developing new SBP algorithms specifically for humanoids, we focus on enabling the standard SBP algorithms to solve humanoids motion planning problems by modifying the underlying key components of generic SBP approaches, such as *space representation*, *sampling strategies* and *interpolation functions*. In order to make the method generic for any humanoid platforms, rather than store balanced samples during offline processing, we use a non–linear optimization based [15] whole–body inverse kinematics (IK) solver to generate balanced samples on–the–fly. Thus, the proposed method can be easily applied to different humanoid robot platforms without extensive pre–processing and setup. We evaluate the proposed method on a 36 DoF Boston Dynamics Atlas and a 38 DoF NASA Valkyrie humanoid robots, to show that our method is capable of generating reliable collision–free whole–body motion for a generic humanoid. We also evaluate the difference between sampling in end–effector and configuration spaces for different scenarios, and compare the planning time and trajectory length to find an optimal trade off between efficiency and optimality. In particular, we apply our work to solve practical reaching tasks on the Valkyrie robot, as highlighted in Fig. 1, showing that the proposed method can generate reliable whole–body motion that can be executed on full–size humanoid robots.

II. Problem Formulation

Let $\mathcal{C} \in \mathbb{R}^{N+6}$ be a robot's configuration space with N the number of articulated joints and the additional 6-DoF of the under actuated virtual joint that connects the robot's pelvis (T_{pelvis}) and the world $\mathcal{W} \in SE(3)$. Let $\mathbf{q} \in \mathcal{C}$ be the robot configuration state, $\mathcal{C}_{balance} \subset \mathcal{C}$ the manifold that contains statically balanced configurations, $\mathcal{C}_{free} \subset \mathcal{C}$ the manifold contains collision free configurations and $\mathcal{C}_{valid} \equiv \mathcal{C}_{balance} \cap \mathcal{C}_{free}$ the valid configuration manifold.

For humanoid robots, valid trajectories can only contain states from valid configuration manifold, i.e. $\mathbf{q}_{[0:T]} \subset \mathcal{C}_{valid}$. Generating collision free samples is straightforward by using random sample generators and standard collision checking libraries. However, generating balanced samples is non-trivial, where a random sampling technique is incapable of efficiently finding balanced samples on the low dimensional manifold $\mathcal{C}_{balance}$ by sampling in high dimensional configuration space $\mathcal{C}$. Guided sampling or pre–sampling process is required for efficient valid sample generation. In our approach, a whole-body inverse kinematic solver is employed to produce statically balanced samples. Static balance constraint is a combination of feet and CoM poses constraints, i.e. the static balance constraint is considered as satisfied when the robot's feet has stable contacts with ground and the CoM ground projection stays within the support polygon.

A. Whole–body Inverse Kinematics

Given a seed configuration $\mathbf{q}_{seed}$ and nominal configuration $\mathbf{q}_{nominal}$ and a set of constraints $\mathbf{C}$, an output configuration that satisfies all the constraints can be generally formulated as:

$$\mathbf{q}^* = IK(\mathbf{q}_{seed}, \mathbf{q}_{nominal}, \mathbf{C}) \tag{1}$$

The Constraint set for a whole–body humanoid robot may include single joint constraints, such as position and velocity limits for articulated joints, it may also include workspace pose constraints, e.g. end–effector poses, centre-of-mass position. In the rest of the paper, unless specified otherwise, we assume the quasi–static balance constraint and joint limits constraints are included in $\mathbf{C}$ by default. We formulate the IK problem as a non-linear optimization problem (NLP) of form:

$$\mathbf{q}^* = \arg\min_{\mathbf{q}\in\mathbb{R}^{N+6}} \|\mathbf{q} - \mathbf{q}_{nominal}\|^2_{Q_q} \tag{2}$$

$$\text{subject to} \quad \mathbf{b}_l \leq \mathbf{q} \leq \mathbf{b}_u \tag{3}$$

$$c_i(\mathbf{q}) \leq 0, c_i \in \mathbf{C} \tag{4}$$

where $Q_q \succeq 0$ is the weighting matrix, $\mathbf{b}_l$ and $\mathbf{b}_u$ are the lower and upper joint bounds. We use a randomly sampled state as the seed pose $\mathbf{q}_{seed}$. We then use this pose as the initial value in the first iteration of SQP solver. Depending on the implementation of the SBP algorithm, we either choose $\mathbf{q}_{nominal}$ to be the current robot state or one of the neighbouring poses drawn from the pool of candidate poses already explored by the SBP algorithm.

B. Sampling–based Motion Planning

Let $\mathbf{x} \in \mathcal{X}$ be the space where the sampling is carried out. The planning problem can be formulated as

$$\mathbf{q}_{[0:T]} = Planning(\mathbf{Rob}, \mathbf{Env}, \mathbf{x}_0, \mathbf{x}_T) \tag{5}$$

where $\mathbf{Rob}$ is the robot model and $\mathbf{Env}$ is the environment instance in which this planning problem is defined. $\mathbf{x}_0$ and $\mathbf{x}_T$ are the initial and desired states.

In order for SBP algorithms to be able to plan motions for humanoid robots, we need to modify the following components that are involved in most algorithms as shown in Fig. 2: the space $\mathcal{X}$ where the sampling is carried out; the strategies to draw random samples; and the interpolation function which is normally used in steering and motion evaluation steps. In the next section, we will discuss the details of modifications we applied on those components for scaling standard SBP algorithms to humanoids.

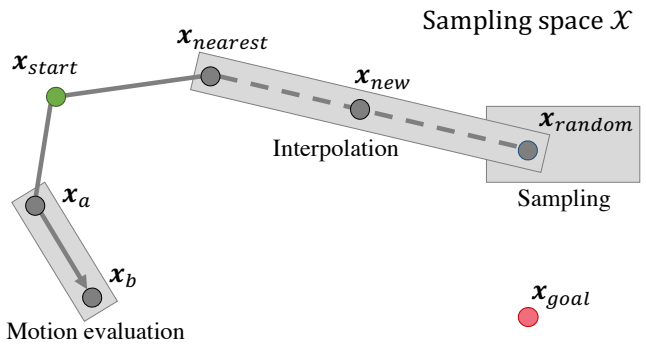


Fig. 2: Instead of developing new algorithms, we modify those underlying components in SBP solvers to make standard algorithms be capable of solving motion planning problems for humanoid robots.

III. Sampling–based Planning for Humanoids

We separate the work into two parts, configuration space sampling and end–effector space sampling. In configuration space sampling approach, the state is represented in $\mathbb{R}^{N+6}$ space with joint limits and maximum allowed base movement as the bounds, the sampling state is identical to robot configuration, i.e. $\mathbf{x} = \mathbf{q} \in \mathcal{C}$. For reaching and grasping problems, one might be interested in biasing the sampling in the end–effector related constraints, e.g. to encourage shorter end–effector traverse distance. The end–effector space approach samples in $SE(3)$ space with a region of interests around the robot as the bounds, the state is equivalent to the end–effector's forward kinematics, i.e. $\mathbf{x} = \Phi(\mathbf{q}) \in \mathcal{W}$ where $\Phi(\cdot)$ is the forward kinematics mapping. However, the final trajectories are represented in configuration space, thus we associate a corresponding configuration for each end–effector space state to avoid ambiguity and duplicated calls of IK solver.

A. Configuration Space Sampling

Algorithm 1 highlights the components' modifications required for sampling in configuration space:

1) Sampling Strategies: For $sampleUniform()$, we first generate random samples from $\mathcal{X}$ and then use fullbody IK solver to process the random samples to generate samples from the balanced manifold $\mathcal{X}_{balance}$

$$\mathbf{q}_{rand} = IK(\bar{\mathbf{q}}_{rand}, \bar{\mathbf{q}}_{rand}, \mathbf{C}) \tag{6}$$

where $\bar{\mathbf{q}}_{rand} \in \mathcal{X}$ is a uniform random configuration and $\mathbf{q}_{rand} \in \mathcal{X}_{balance}$ is random sample from the balanced manifold. We use $\bar{\mathbf{q}}_{rand}$ as nominal pose since we want to generate random postures rather than postures close to other already existing samples. This is to indirectly encourage exploration of the null-space of the task. The constraint set $\mathbf{C}$ contains static balance constraint and joint limits constraints. When sampling around a given state, $sampleUniformNear(\mathbf{q}_{near}, d)$, we first get a random state $\bar{\mathbf{q}}_{rand}$ that is close to $\mathbf{q}_{near}$ within distance d. The IK solver is invoked with $\bar{\mathbf{q}}_{rand}$ as the seed pose, and $\mathbf{q}_{near}$ as the nominal pose. An additional configuration space constraint is added to the constraint set

$$\|\mathbf{q}_{rand} - \bar{\mathbf{q}}_{rand}\|_W \leq A \tag{7}$$

where $A \in \mathbb{R}^{N+6}$ is a tolerance vector initially set to zero. In most cases the system will be over constrained, in which case we need to increase the tolerance to ensure balance. Normally, the lower–body joints are neglected first, i.e. increasing corresponding w_i, meaning that we allow the lower–body joints to deviate from $\bar{\mathbf{q}}_{rand}$ in order to keep feet on the ground and maintain balance. We use $\mathbf{x}_{near}$ as the nominal pose since later on the random state is likely to be appended to $\mathbf{q}_{near}$, where one wants the random state be close to the near state. The new sample is discarded if the distance between $\mathbf{q}_{near}$ and $\mathbf{q}_{rand}$ exceeds the limit d.

2) Interpolation: In order to find a balanced state interpolated along two balanced end–point states, we first find the interpolated, likely to be un–balanced state

$$\bar{\mathbf{q}}_{int} = \mathbf{q}_a + d\|\mathbf{q}_b - \mathbf{q}_a\|. \tag{8}$$

A similar configuration space constraint to (7) is applied to constrain the balanced interpolated state $\mathbf{q}_{int}$ close to $\bar{\mathbf{q}}_{int}$

$$|\mathbf{q}_{int} - \bar{\mathbf{q}}_{int}\| \leq A \tag{9}$$

Algorithm 1 Humanoid Configuration Space SBP

sampleUniform()

$succeed$ = False
while not $succeed$ **do**
  $\bar{\mathbf{q}}_{rand} = RandomConfiguration()$
  $\mathbf{q}_{rand}, succeed = IK(\bar{\mathbf{q}}_{rand}, \bar{\mathbf{q}}_{rand}, \mathbf{C})$
return $\mathbf{q}_{rand}$

sampleUniformNear($\mathbf{q}_{near}, d$)

$succeed$ = False
while not $succeed$ **do**
  $A \leftarrow Zeros(N+6)$
  while not $succeed$ **do**
    $\bar{\mathbf{q}}_{rand} = RandomNear(\mathbf{q}_{near}, d)$
    Set constraint $\|\mathbf{q}_{rand} - \bar{\mathbf{q}}_{rand}\|_W < A$
    $\mathbf{q}_{rand}, succeed = IK(\bar{\mathbf{q}}_{rand}, \mathbf{q}_{near}, \mathbf{C})$
    Increase A
  if $distance(\mathbf{q}_{rand}, \mathbf{q}_{near}) > d$ **then**
    $succeed$ = False
return $\mathbf{q}_{rand}$

interpolate($\mathbf{q}_a, \mathbf{q}_b, d$)

$\bar{\mathbf{q}}_{int} = InterpolateConfigurationSpace(\mathbf{q}_a, \mathbf{q}_b, d)$
$succeed$ = False
$A \leftarrow Zeros(N+6)$
while not $succeed$ **do**
  Set constraint $\|\mathbf{q}_{int} - \bar{\mathbf{q}}_{int}\|_W < A$
  $\mathbf{q}_{int}, succeed = IK(\bar{\mathbf{q}}_{int}, \mathbf{q}_a, \mathbf{C})$
  Increase A
return $\mathbf{q}_{int}$

Algorithm 2 Humanoid End–Effector Space SBP

sampleUniform()

1: $succeed = \text{False}$
2: **while not** $succeed$ **do**
3: $\quad \bar{\mathbf{x}}_{rand} = RandomSE3()$
4: $\quad$ Set constraint $\|\bar{\mathbf{x}}_{rand} - \Phi(\mathbf{q}_{rand})\| \leq 0$
5: $\quad \mathbf{q}_{rand}, succeed = IK(\bar{\mathbf{q}}_{rand}, \bar{\mathbf{q}}_{rand}, \mathbf{C})$
6: $\mathbf{x}_{rand} = \Phi(\mathbf{q}_{rand})$
return $\mathbf{x}_{rand}, \mathbf{q}_{rand}$

sampleUniformNear($\mathbf{x}_{near}, d$)

1: $succeed = \text{False}$
2: **while not** $succeed$ **do**
3: $\quad \bar{\mathbf{x}}_{rand} = RandomNearSE3(\mathbf{x}_{near}, d)$
4: $\quad$ Set constraint $\|\bar{\mathbf{x}}_{rand} - \Phi(\mathbf{q}_{rand})\| \leq 0$
5: $\quad \mathbf{q}_{rand}, succeed = IK(\mathbf{q}_{rand}, \mathbf{q}_{near}, \mathbf{C})$
6: $\mathbf{x}_{rand} = \bar{\mathbf{x}}_{rand}$
return $\mathbf{x}_{rand}, \mathbf{q}_{rand}$

interpolate($\mathbf{x}_a, \mathbf{x}_b, d$)

1: $\bar{\mathbf{x}}_{int} = InterpolateSE3(\mathbf{x}_a, \mathbf{x}_b, d)$
2: $succeed = \text{False}$
3: $B \leftarrow Zeros(SE3)$
4: **while not** $succeed$ **do**
5: $\quad$ Set constraint $\|\bar{\mathbf{x}}_{int} - \Phi(\mathbf{q}_{int})\| < B$
6: $\quad \mathbf{q}_{int}, succeed = IK(\mathbf{q}_a, \mathbf{q}_a, \mathbf{C})$
7: $\quad$ Increase B
8: $\mathbf{x}_{int} = \Phi(\mathbf{q}_{int})$
return $\mathbf{x}_{int}, \mathbf{q}_{int}$

The two end-point states $\mathbf{q}_a$ and $\mathbf{q}_b$ are valid samples generated using our sampling strategies. Due to the convex formulation of the balance constraint, a valid interpolated state is guaranteed to be found. It is worth mentioning that in some cases the interpolation distance equation no longer holds after increasing the tolerance, i.e. $\frac{\|\mathbf{x}_{int} - \mathbf{x}_a\|}{\|\mathbf{x}_b - \mathbf{x}_a\|} \neq d$. However, this is a necessary step to ensure that the balance constraint are satisfied.

B. End-Effector Space Sampling

Algorithm 2 highlights the components' modifications required for sampling in end–effector space:

1) Sampling Strategies: It is straight forward to sample in $SE(3)$ space, however, it is non–trivial to sample balanced samples from the $\mathcal{X}_{balance}$ manifold. For $sampleUniform()$, we first randomly generate $SE(3)$ state $\bar{\mathbf{x}}_{rand}$ within a region of interest in front of the robot. The whole–body IK is invoked with an additional end–effector pose constraint

$$\|\bar{\mathbf{x}}_{rand} - \Phi(\mathbf{q}_{rand})\| \leq 0 \tag{10}$$

The sampler keeps drawing new random states $\bar{\mathbf{x}}_{rand}$ until the SQP solver returns a valid output $\mathbf{q}^*$. The valid random state $\mathbf{x}_{rand}$ can be calculated using forward kinematics, e.g. $\mathbf{x}_{rand} = \Phi(\mathbf{q}^*)$. The same procedure applies to $sampleNear(\mathbf{x}_{near}, d)$, but using $\mathbf{x}_{near}$ as the seed configuration.

2) Interpolation: Similar to sampling near a given state, for interpolation in end–effector space, we first find the interpolated state $\bar{\mathbf{x}}_{int} \in SE(3)$ and add the following term into constraint set

$$\|\bar{\mathbf{x}}_{int} - \Phi(\mathbf{q})\| \leq B \tag{11}$$

where $B \in \mathbb{R}^6$ is a tolerance vector initially set to zero. If the system is over constrained after adding end–effector pose constraint, we need to selectively relax the tolerance for different dimensions $(x, y, z, roll, pitch, yaw)$ until the IK solver succeeds. Then we reassign the interpolated state using forward kinematics, $\mathbf{x}_{int} = \Phi(\mathbf{q}_{int})$.

3) Multi-Endeffector Motion Planning: Some tasks require coordinated motion involving multiple end–effectors, e.g. bi–manual manipulation and multi–contact motion. It is obvious that, from a configuration space point of view, there is no difference as long as the desired configuration is specified. It is also possible for end–effector space sampling approach to plan motion with multiple end–effector constraints. Let $\mathbf{y}_k^* \in SE(3)$ be the desired pose constraints for end–effector $k \in \{1, \ldots, K\}$. A meta end–effector space $\mathcal{X} \in \mathbb{R}^{6 \times K}$ can be constructed to represent the sampling space for all end–effectors. Similar sampling and interpolation functions can be implemented by constructing extra constraints for each end–effector k.

TABLE I: Planning time of empty space reaching problem crossing different algorithms, in seconds.

Algorithms	Sampling Space	
	End–Effector Space	Configuration Space
RRT	25.863 ± 22.894	1.4129 ± 1.4466
PRM	4.2606 ± 3.0322	0.5912 ± 0.5912
EST	28.055 ± 18.270	0.3112 ± 0.3112
BKPIECE	5.3989 ± 5.9470	0.1781 ± 0.0332
SBL	3.0602 ± 0.9859	0.2804 ± 0.0480
RRT–Connect	2.8228 ± 0.3412	0.1853 ± 0.0450

IV. Evaluation

We aim to generalize the common components of sampling–based motion planning algorithms for humanoid robots so that existing algorithms can be used without extra modification. We implemented our approach in the EXOTica motion planning and optimization framework [16] as humanoid motion planning solver, which internally invokes the SBP planners from the Open Motion Planning Library (OMPL, [17]). We have set up the system with our customized components, and evaluated our approach on the following 6 representative algorithms: RRT [5], RRT-Connect [7], PRM [6], BKPIECE [10], EST [8]) and SBL [18]. The evaluations are performed on a single thread of the 4.0 GHz Intel Core i7-6700K CPU.

A. Empty Space Reaching

In the first experiment, we have the robot reach a target pose in front of the robot in free space, where only self–collision and balance constraints are considered. This is a

TABLE II: Evaluation of whole–body collision–free motion planning. RRT–Connect$_e$ sampling in end–effector space, all other methods sampling in configuration space. $\mathcal{C}$ cost is the configuration space trajectory length, $\mathcal{W}$ cost is the end–effector traverse distance in workspace, CoM cost is the CoM traverse distance in workspace. No. evaluation shows the number of state evaluation calls, i.e. evaluate if a sampled/interpolated state is valid. No. IK indicates the number of online whole–body IK calls, and IK time is the total time required for solving those IK, which is the most time consuming element. The result is averaged over 100 trails.

Tasks	Algorithms	Planning time (s)	$\mathcal{C}$ cost (rad.)	$\mathcal{W}$ cost (m)	CoM cost (m)	No. evaluation	No. IK	IK time (s)
Task 1	BKPIECE$_c$	42.5 ± 26.4	7.37 ± 2.43	2.10 ± 0.80	0.24 ± 0.10	1946 ± 1207	2598 ± 1582	41.4 ± 25.7
	SBL$_c$	27.8 ± 8.59	6.25 ± 1.06	2.14 ± 0.71	0.23 ± 0.06	1313 ± 418	1508 ± 445	27.0 ± 8.33
	RRT–Connect$_e$	9.91 ± 4.80	2.93 ± 0.96	**0.58** ± 0.11	**0.07** ± 0.02	597 ± 354	727 ± 387	9.51 ± 4.58
	RRT–Connect$_c$	**1.53** ± 0.80	**2.71** ± 0.68	0.99 ± 0.23	0.11 ± 0.03	**95** ± 54	**118** ± 64	**1.48** ± 0.77
Task 2	BKPIECE$_c$	40.5 ± 21.7	6.59 ± 2.43	1.95 ± 0.59	0.27 ± 0.09	1911 ± 970	2473 ± 1254	39.4 ± 20.1
	SBL$_c$	22.2 ± 9.51	5.34 ± 2.00	1.79 ± 0.80	0.24 ± 0.09	1089 ± 472	1259 ± 547	21.5 ± 9.23
	RRT–Connect$_e$	12.4 ± 6.65	4.12 ± 2.02	**0.77** ± 0.08	**0.09** ± 0.04	656 ± 405	826 ± 458	11.9 ± 6.41
	RRT–Connect$_c$	**2.25** ± 0.85	**3.29** ± 1.14	1.20 ± 0.33	0.14 ± 0.05	**106** ± 42	**166** ± 59	**2.19** ± 0.83
Task 3	BKPIECE$_c$	45.7 ± 19.8	7.49 ± 2.52	1.96 ± 0.73	0.25 ± 0.08	2057 ± 949	2758 ± 1166	44.5 ± 19.3
	SBL$_c$	33.8 ± 22.2	8.68 ± 2.26	2.10 ± 0.44	0.28 ± 0.11	1414 ± 950	1756 ± 1151	33.0 ± 21.6
	RRT–Connect$_e$	25.3 ± 13.9	7.19 ± 4.93	**0.92** ± 0.13	0.16 ± 0.05	1031 ± 532	1436 ± 720	24.6 ± 13.7
	RRT–Connect$_c$	**3.45** ± 0.77	**4.68** ± 0.59	1.38 ± 0.12	**0.14** ± 0.03	**165** ± 49	**200** ± 53	**3.36** ± 0.75

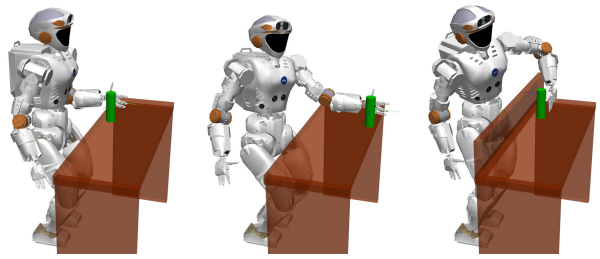
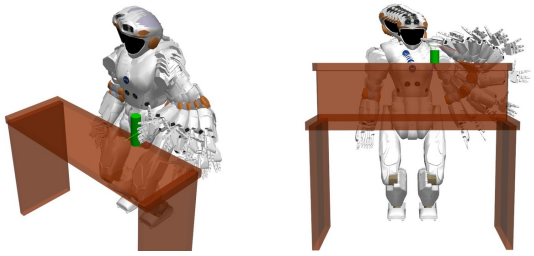
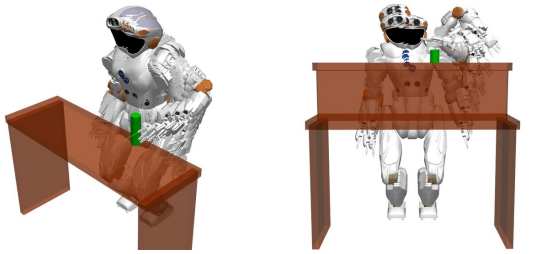


Fig. 3: Evaluation tasks, from left to right: task 1, target close to robot; task 2, target far away from robot; and task 3, target behind bar obstacle.



(a) Trajectories generated using configuration space sampling.



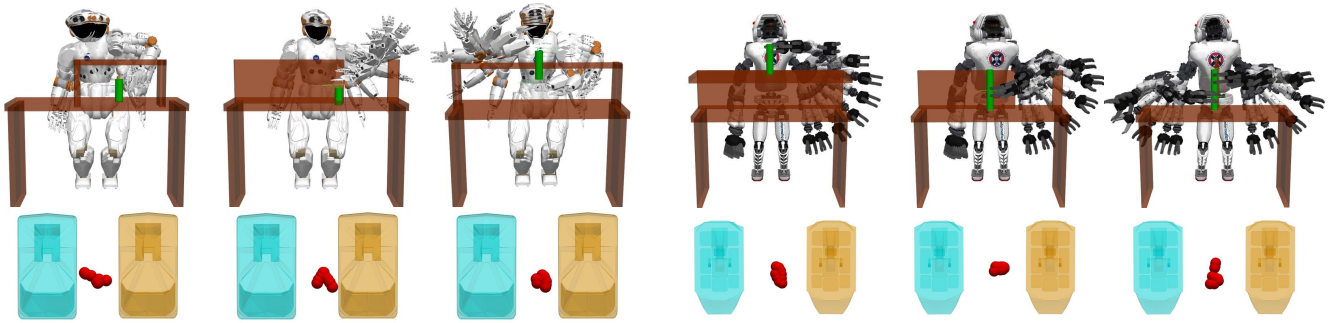
(b) Trajectories generated using end–effector space sampling.

Fig. 4: Whole–body motion plans generated using different sampling spaces. The task is identical for each column. In general, configuration space sampling leads to shorter trajectory length; end–effector space sampling leads to shorter end–effector traverse distance.

sanity check to show that the proposed method can be used robustly across different planning algorithms to generate trajectories for humanoid robots. We solve the reaching problem using the 6 testing algorithms in two different sampling spaces, each across 100 trials. The results are shown in Table I. Although the planning time varies across different algorithms and sampling spaces, the result shows that standard planning algorithms are able to generate motion plans for humanoid robots using our method. However, as expected, bi–directional algorithms are more efficient than their unidirectional variants. Also, sampling in configuration space is much more efficient than in end–effector space due to the higher number of IK calls.

B. Collision–free Reaching

We setup three different scenarios, from easy to hard, as illustrated in Fig. 3, to evaluate the performance of different algorithms in different sampling spaces. Unfortunately, the evaluation suggests that standard unidirectional algorithms are unable to solve these problems (within a time limit of 100 seconds). Without bi–directional search, the high dimensional humanoid configuration space is too complex for sampling–based methods to explore. Table II highlights the results using four different bidirectional approaches. Note that when sampling in end–effector space, only RRT–Connect is able to find a valid solution in the given time, other bidirectional search algorithms like BKPIECE and SBL are also unable to find valid trajectories. The result indicates that RRT–Connect sampling in configuration space is the most efficient and the most robust approach for solving humanoid whole–body motion planning problems. It requires the least exploration, thus bypassing expensive online IK queries. Algorithms like BKPIECE and SBL use low–dimensional projections to bias the sampling, however, the default projections which are tuned for mobile robots and robotic arms do not scale up to high DoF humanoid robots, which leads to long planning time and trajectories



(a) Reaching motion on the NASA Valkyrie robot.

(b) Reaching motion on the Boston Dynamics Atlas robot.

Fig. 5: Collision–free whole–body motion generated in different scenarios with different robot models. The corresponding CoM trajectories are illustrated in the second row (red dots). The framework is setup so that one can easily switch to new robot platforms without extensive preparing procedures.

with high costs. This can be improved by better projection bias, but it is non–trivial to find a suitable bias without fine tuning. Also, the trajectories generated using RRT–Connect are shorter, meaning that the motion is more stable and robust. It is worth mentioning that RRT–Connect takes longer time to plan when sampling in the end–effector space than it does in the configuration space, but the planned trajectories have shorter end–effector and CoM traverse distances. In some scenarios where planning time is not critical, one choose to use RRT–Connect in end–effector space to generate trajectories with shorter end–effector traverse distance. These results also suggest that the whole–body IK computation dominates the planning time. This is in contrast with classical SBP problems where collision–detection is the the most time consuming component. However, the IK solver is necessary for keeping balance, as shown in Fig. 5, where the trajectories' CoM projections are within the support polygon.

In more complex scenarios, such as reaching through narrow passages and bi–manual tasks, most algorithms fail to generate valid trajectories apart from RRT–Connect. As mentioned, some algorithms' performance depends on the biasing methods, e.g. projection bias and sampling bias. However, it is non–trivial to find the appropriate bias for humanoids that would generalize across different tasks. Fig. 5 highlights some examples of reaching motion in more complex scenarios with different robot models. As stated earlier, this work focuses on generalising SBP algorithms for humanoids, where as one can easily setup the system on new robot platforms. For instance, one can easily switch from Valkyrie (Fig. 5a) to Atlas (Fig. 5b) in minutes without extensive pre–processing and setup procedures.

In order to test the reliability and robustness of the proposed method, we applied our work on the Valkyrie robot accomplishing reaching and grasping tasks in different scenarios, as highlighted in Fig 6. During practical experiments, the collision environment is sensed by the on–board sensor and represented as an octomap [19]. The experiment results show that our method is able to generate collision–free whole–body motion plans that can be executed on full–size humanoid robot to realise practical tasks such as reaching and grasping. A supplementary video of the experiment results can be found at `https://youtu.be/W48miMKWnW4`.

V. Conclusion

In this paper we generalise the key components required by sampling–based algorithms for generating collision–free and balanced whole–body trajectories for humanoid robots. We show that by using the proposed methods, standard algorithms can be invoked to directly plan for humanoid robots. We also evaluate the performance of different algorithms on solving planning problems for humanoids, and point out the limitations of some algorithms. A variety of different scenarios are tested showing that the proposed method can generate reliable motion for humanoid robots in different environments. This work can be transferred to different humanoid robot models with easy setup procedure that can be done in very a short period of time, without extensive pre-computation for adapting the existing algorithms to different robot models, as we have tested on the 36 DoF Boston Dynamics Atlas and the 38 DoF NASA Valkyrie robots. In particular, we applied this work on the Valkyrie robot accomplishing different tasks, showing that the proposed method can generate robust collision–free whole–body motion that can be executed on real robots.

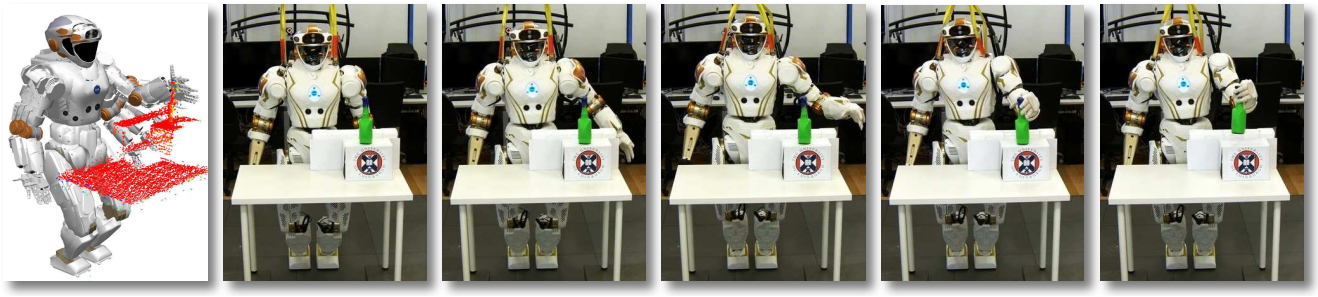
The result in Table II shows that the whole–body IK solver dominates over 95% of the online computation time, which currently only runs on a single–thread but can be parallelised on multi–threaded CPU/GPU. The future work will include investigating parallelised implementation of the IK solver on GPU to bootstrap sampling and interpolation. This will make the state space exploration more efficient, so that other standard algorithms may be able to find valid solutions within the same time window.

References

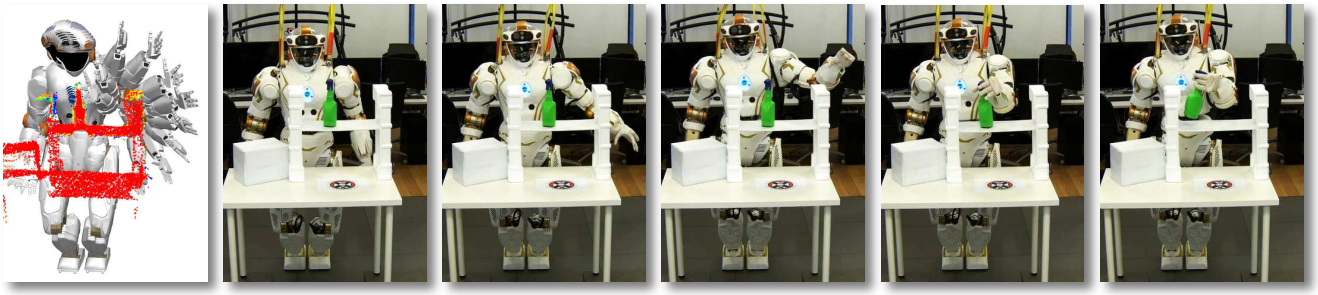
[1] K. Rawlik, M. Toussaint, and S. Vijayakumar, "On stochastic optimal control and reinforcement learning by approximate inference," *RSS*, 2012.



(a) Reach and grasp target on table without facing target.



(b) Reach and grasp target on top of box.



(c) Reach and grasp target on top of shelf.

Fig. 6: Collision–free whole–body motion execution on the NASA Valkyrie humanoid robot. In each scenario, the first figure highlights the motion plan, followed by execution snapshots.

[2] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *ICRA, IEEE*, 2009.

[3] Y. Yang, V. Ivan, and S. Vijayakumar, "Real-time motion adaptation using relative distance space representation," in *ICAR, IEEE*, 2015.

[4] Y. Yang, V. Ivan, Z. Li, M. Fallon, and S. Vijayakumar, "iDRM: Humanoid Motion Planning with Real-Time End-Pose Selection in Complex Environments," in *Humanoids (submitted to)*, 2016.

[5] S. M. Lavalle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning," tech. rep., 1998.

[6] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *ICRA*, 1996.

[7] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *ICRA, IEEE*, 2000.

[8] D. Hsu, J. C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," in *ICRA, IEEE*, 1997.

[9] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *IJRR*, pp. 846–894.

[10] I. A. Şucan and L. E. Kavraki, "Kinodynamic motion planning by interior-exterior cell exploration," in *Algorithmic Foundation of Robotics*, 2009.

[11] M. Elbanhawi and M. Simic, "Sampling-Based Robot Motion Planning: A Review," *IEEE Access*, 2014.

[12] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion planning for humanoid robots," in *ISRR*, 2005.

[13] K. Hauser, T. Bretl, K. Harada, and J.-C. Latombe, "Using motion primitives in probabilistic sample-based planning for humanoid robots," in *Algorithmic foundation of robotics*, 2008.

[14] M. Cognetti, P. Mohammadi, and G. Oriolo, "Whole-body motion planning for humanoids based on CoM movement primitives," in *Humanoids, IEEE*, 2015.

[15] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM*, 2005.

[16] V. Ivan, Y. Yang, and M. Camilleri, "EXOTica: a library for easy creation of tools for optimisation and planning," 2016.

[17] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *Robotics Automation Magazine, IEEE*, 2012.

[18] G. Sánchez and J.-C. Latombe, "A single-query bi-directional probabilistic roadmap planner with lazy collision checking," in *Robotics Research*, 2003.

[19] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees," *Autonomous Robots*, 2013.