Motion Adaptation for Humanoid Robots in Constrained Environments

Edmond S. L. Ho and Hubert P. H. Shum

***Abstract*—This paper presents a new method to synthesize full body motion for controlling humanoid robots in highly constrained environments. Given a reference motion of the robot and the corresponding environment configuration, the spatial relationships between the robot body parts and the environment objects are extracted as a representation called the *Interaction Mesh*. Such a representation is then used in adapting the reference motion to an altered environment. By preserving the spatial relationships while satisfying physical constraints, collision-free and well balanced motions can be generated automatically and efficiently. Experimental results show that the proposed method can adapt different full body motions in significantly modified environments. Our method can be applied in precise robotic controls under complicated environments, such as rescue robots in accident scenes and searching robots in highly constrained spaces.**

I. Introduction

Controlling humanoid robots to work in constrained environments is a challenging task. Imagine scenarios such as searching for survivors in a ruin or navigating in a factory after an explosion, the movements of the robot must be precisely controlled due to the potential collision with surrounding obstacles. Global path planning is one of the most commonly used motion synthesis algorithms for controlling humanoid robots nowadays. Given the start and goal configurations of a robot, such as the 3D locations or joint angles of the body parts, a collision-free motion is computed in the configuration space. In early works [1], Rapidly-exploring Random Tress (RRT) had been applied for solving various problems including 2D path planning and a 6-DOF PUMA arm motion planning. Since then, researchers successfully extended RRT to create collision-free motions for controlling humanoid robots [2], [3], [4]. Yamane et al. [5] synthesized character motions like moving objects in a 3D virtual environment.

While the effectiveness of RRT has been verified by numerous researches, its major shortcoming lies in the computational complexity. With the increase of the number of DOF, the computation cost rises exponentially. This drawback becomes more significant when planning the full body motion of a humanoid robot in constrained environments, because the randomly sampled configurations are likely to cause collisions and will be discarded.

The fundamental problem lies in the representation of the robot and the surrounding objects in the configuration space. In traditional motion planning algorithms, the configuration space separately encodes the information of the robot and the objects, such as their respective positions and orientations, instead of their relative relationships. Sampling movements in such a representation usually results in a large number of collisions, as there is no explicit quantity to describe how close the robot's parts are to the surroundings. For the same reason, it is also very difficult for the robot to maintain a relative distance with the surrounding objects throughout a movement, especially in complex environments.

In order to reduce the dimensionality of the configuration space for efficient encoding of the spatial relationships between the robot and the environment, Ho et al. [6] proposed to use Topology Coordinates [7] for synthesizing close interactions such as twisting the limbs between two Nao robots [8] in real-time. Topology Coordinates is an abstract representation and has been applied to efficiently synthesize various kinds of complex motions and manipulations such as interacting with non-rigid materials like a t-shirt [9], [10] and regrasping movements [11]. The major limitation of the Topology Coordinates is that it can only represent the spatial relationships of motions that involve tangles. In addition, the volume information of the robot's body segments is not considered, making it unable to guarantee collision-free movements.

To tackle this problem, Ho et al. [12] proposed the Interaction Mesh to represent the spatial relationships of a character with its surrounding environment or another character. It is, however, unclear how the method can be extended to create physically valid movements for robotic controls. Zarubin et al. [13] attempted to combine topological representation including Topology Coordinates and Interaction Mesh, as well as low-level information such as the location of the end-effector, for robotic planning. The method had been successfully applied for planning the motion for reaching tasks on the KUKA LWR 4 robotic arm in a dynamic environment. In this research, we focus on adapting full-body motions to the changes of constrained environments.

In this paper, we propose a new method to synthesize full body motion of humanoid robots adapting to the changes of the environment (Fig. 4 and 5). To facilitate efficient planning, we utilize Interaction Mesh [12] to encode the spatial relationship between body segments and nearby objects into the configuration space representation. Given a reference motion of a robot and its corresponding environment, we can synthesize a new motion in an altered environment by minimizing the distortion of the Interaction Mesh. Planning

E. S. L. Ho is with Department of Computer Science, Hong Kong Baptist University. Email: edmond@comp.hkbu.edu.hk

H. P. H. Shum is with Faculty of Science and Engineering, Northumbria University, UK. Email: hubert.shum@northumbria.ac.uk

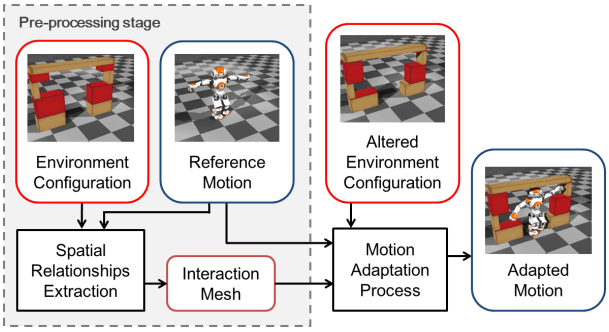

Fig. 1. The overview of our proposed method.

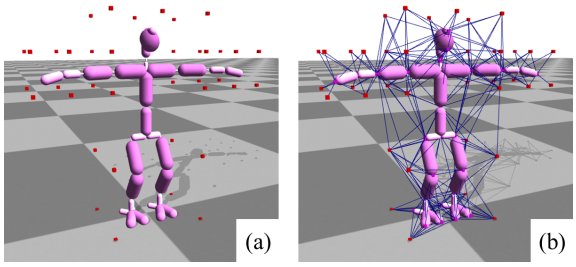

Fig. 2. An example of the Interaction Mesh computed from the vertices sampled from the robot and the objects in the environment. (a) The sampled vertices. (b) The Interaction Mesh computed.

with the Interaction Mesh is superior to traditional algorithms in many aspects. First, operators do not have to define/guide the target trajectories of the joints, as the mesh extracts implicit relationships from examples automatically. Second, it reduces the complexity of the motion synthesizing problem by abstracting major relationships.

We demonstrate our algorithm by controlling the Nao robot under highly constrained situations. We show that our system can automatically extract implicit relationship between the robot and the environment from an example, and synthesize dynamically stable movement in altered environments without guidance from the operator. Our method can be applied in precise robotic controls, especially for humanoid robots that have high Degree of Freedom, in complicated environments such as accident scenes and complex spaces.

Here are the major contributions of this paper:

- We redesign the energy functions proposed to enable the Interaction Mesh framework to compute joint angles directly, as oppose to joint positions in the Cartesian coordinates shown in [12]. This allows direct control over the joints of the humanoid robots.
- We propose additional physical constraints on top of the kinematic constraints designed in [12] in the motion adaptation process to preserve physical correctness of the synthesized motions. This ensures that the humanoid robots controlled by the resultant motion maintain balance throughout the movement.

II. Overview

The overview of our proposed method is shown in Fig. 1. In the preprocessing stage, given a reference motion of the humanoid robot and the configurations of the objects in the environment, our method extracts their spatial relationships as an Interaction Mesh (Section III). During run-time, when the environment has changed, the Interaction Mesh will be used in the motion adaptation process (Section IV) to preserve the context of the motion subject to various dynamics and kinematics constraints (Section V). The adapted motion is a physically valid motion that fits into the new environment.

III. Spatial Relationships Extraction

In this section, we explain how we extract the relationship between the robot and its surrounding environment.

During the pre-processing stage, we compute the Interaction Mesh based on a reference robot motion and the corresponding configuration of the environment. The Interaction Mesh is a volumetric mesh composed of vertices and edges. The vertices consist of (1) the locations of the robot's joints, which are directly obtained from the robot's kinematic configuration, and (2) the points sampled from the surfaces of the objects in the environment. To facilitate efficient computation during the motion adaptation process, the number of vertices representing the object should be minimal. In our implementation, we sample the vertices at the corners of the objects, which are assumed to be the extreme positions. Surfaces are sampled with a uniform grid. Empirically, we found that a resolution of 20cm results in sufficiently accurate representation. We then apply Delaunay Tetrahedralization [14] to compute a volumetric mesh by considering all vertices as a single point cloud. Since the vertices close to each other are connected by edges, the body segments and the objects that are closely interacting with each other can be modelled. Fig. 2 (a) shows the sampled vertices computed from a given scenario, and Fig. 2 (b) shows the corresponding Interaction Mesh.

We use the joints as the vertexes of the robot based on the assumption that the body parts can be well represented by the joints. In case the body parts contain unusual shapes or dimensions, we may sample vertexes on the surface of the parts. On the other hand, if the environment contains a large number of objects in irregular shapes, the sampling process can be performed automatically with mesh simplification techniques such as Triangular Mesh Decimation [15]. Given the 3D geometry mesh of the environment, mesh simplification techniques simplify the environment by removing redundant vertices while preserving the underlying features. The reduced set of vertices can be used to compute the Interaction Mesh efficiently.

IV. Environment Changes Adaptation

In this section, we present the method for editing the motion of the humanoid robot to adapt the changes of the environment.

Our goal is to adjust the reference motion to produce collision-free motion in the updated environment, while preserving the spatial relationships between the robot's body parts and the objects in the environment. By maintaining the

topology of the Interaction Mesh and minimizing its distortion, such relationships can be preserved in the motion adaptation process. We formulate this motion adaptation process as an optimization problem, which is modelled as an overdetermined system of linear equations with linear constraints. Similar idea has shown to be effective in editing multiple characters interaction [16].

The whole motion is adapted in a single spacetime constraints optimization. The major advantage of this approach, as oppose to per-frame motion editing methods, such as inverse kinematics, is the ability to avoid high frequency movements. In addition, physically-valid motions can be synthesized by incorporating balance-related constraints.

In the following, we will first define different energy functions, and then explain how to optimize the movements. We assume m to be the number of 3D vertices in the interaction mesh, p_j^i where $1 \leq j \leq m$ to be the vertices at frame i and index j, V_i to be a vector of size $3m$ that composed of all p_j^i such that $V_i = ({p_1^i}^\intercal, \cdots, {p_m^i}^\intercal)$, ${p_j^i}'$ and V_i' to be the updated vectors after the deformation, q_i and $\dot{q_i}$ to be the set of the joint angles of the robot and its derivatives respectively.

A. Deformation Energy

The purpose of introducing the deformation energy is to maintain the spatial relationships between the body parts of the robot and the environment by minimizing the distortion of the Interaction Mesh in the motion adaptation process. Although the deformation energy term has been proposed in [12], it is not applicable for controlling humanoid robots as we have to manipulate the joint angles in this research. For this reason, the modified deformation energy function is defined as:

$$E_L(\dot{q_i}) = \sum_j \|\delta_j - L(J_{pos,i}^j \dot{q_i} + {p_j^i}')\|^2 \tag{1}$$

where δ_j is the original Laplacian coordinate, $J_{pos,i}^j$ is the Jacobian of the position derivative for vertex ${p_j^i}'$ with respect to the joint angle derivatives. The Laplacian coordinates are calculated as:

$$L(p_j) = p_j - \sum_{l \in N_j} w_l^j p_l \tag{2}$$

where N_j is the one-ring neighbourhood of p_j, w_l^j representing the normalized weights that are set to be inversely proportional to the distance between the vertices. As a result, farther apart vertices have less influence on each other.

B. Velocity Energy

Interaction Mesh is a frame-based representation. The motion adapted based on a series of Interaction Mesh may contain jittery movements across frames, which affect the stability of the robot. To tackle the problem, the velocity energy term E_V is introduced to imposes temporal relationships between corresponding vertices in adjacent frames. Specifically, we minimize the movement of the corresponding vertices in current and previous frame to reduce sudden acceleration:

$$E_V(\dot{q_i}, \dot{V_{robot,i-1}}) = \|J_{vel,i}\dot{q_i} - \dot{V_{robot,i-1}}\|^2 \tag{3}$$

where $\dot{V_{robot,i-1}}$ is the set of derivatives of the positions of the vertices sampled from the robot at previous frame, $J_{vel,i}$ is the Jacobian of the position derivatives for the vertices sampled from the robot at current frame with respect to the joint angle derivatives.

C. Constraint Energy

One common shortcoming of Spacetime optimization is the possibility of over-constrained, in which the optimizer may fail to find a suitable posture that fits into all requirements. To remedy this, we allow the operators to specify the constraints can be violated based on their needs. This class of constraints is known as *soft constraints*, for which we design an energy function to minimize the amount of violation. On the other hand, those must be satisfied are considered as *hard constraints* and are enforced during the optimization loop as explained in the Section V.

Given a system of linear equations representing the soft constraints, $F_i\dot{q_i} = f_i$, the energy term to represent the amount of violation is defined as:

$$E_C(\dot{q_i}) = \frac{1}{2}\dot{q_i}^\intercal {F_i}^\intercal W F_i \dot{q_i} - {f_i}^\intercal W F_i \dot{q_i} + \frac{1}{2}{f_i}^\intercal W f_i. \tag{4}$$

where W is a square diagonal matrix that assigns a different weight to each constraint.

We use constraints to maintain physical stability and kinematics features. We design four constraints to represent the requirements on collision avoidance, position control, balance and stepping pattern. All of them can be switched to soft or hard during synthesis. The implemented details of individual constraint will be presented in Section V.

D. Iterative Morphing

At every morph step, the positions of the vertices sampled from the environment are gradually updated to the target configuration by linear interpolation as follows:

$$V_{pos}^{int} = \frac{k}{r}(V_{pos}^{target} - V_{pos}^{ref}) + V_{pos}^{ref} \tag{5}$$

where V_{pos}^{ref} and V_{pos}^{target} are the sets of vertices sampled from the reference and altered environment respectively, r is the total number of morph steps, and V_{pos}^{int} contains the interpolated positions of the vertices in the k-th morph step.

The poses of the robot are adapted by minimizing the sum of the deformation (Eq. 1), velocity (Eq. 3) and constraint energy (Eq. 4), subject to the set of hard constraints $H_i\dot{q_i} = h_i$. The adapted motion is computed by solving

$$argmin_{\dot{q_i}, \lambda_i (1 \leq i \leq n)} E_L + w_\Delta E_V + E_C + \lambda_i^\intercal (H_i \dot{q_i} - h_i) \tag{6}$$

where $\dot{q_i}$ is the set of joint angle derivatives at the current frame, λ_i are the Lagrange multipliers and w_Δ is a constant weight set as 0.2. The optimization problem in Eq. 6 can be solved by differentiating it with respect to $\dot{q_i}$ and λ_i, and solving a linear equation. The reader is referred to [12] for further details.

Since we have to gradually change the positions of the vertices sampled from the reference environment to those sampled

from the altered environment, we require a correspondence between the vertices sampled from the two environments. In our implementation, the number of vertices sampled in the two environments is the same, and the operator designs the correspondence manually. In more complex scenarios, we can model the correspondence problem as a mass transport problem [17] and optimize the correspondence by minimizing the sum of distances between all matched vertex pairs.

V. Constraints

In this section, we explain the constraints that can either be used as soft constraints in Section IV-C or hard constraints in Section IV-D. We first introduce the collision and positional constraint adapted from the original Interaction Mesh framework [12]. We then propose the additional balancing and stepping pattern constraint in order to synthesize physically valid motion for robotic controls.

A. Collision Constraints

The collision constraints prevent penetration between the bounding volumes of the skeleton and the objects in the environment. Using the Open Dynamics Engine (ODE) [18], we detect collision of the robot in the updated environment configuration. As in [12], when a penetration is detected, we compute the point pair that penetrates each other the farthest, the penetration depth and direction. We define the collision constraint as:

$$C_C(\dot{q}_i) = J_{collide,i}\dot{q}_i - d_i \tag{7}$$

where $J_{collide,i}$ is the Jacobian of the position derivatives of the colliding parts with respect to the joint angle derivatives, and d_i is the penetration depth multiplied to the normal vectors of the penetrated surface. The Jacobian is computed by finite differencing. The joint angles are changed and the locations of the penetrating points are recomputed according to the posture.

B. Positional Constraints

In order to enable precise control such as reaching the arms to a particular location, positional constraints can be added by anchoring some joints or a linear combination of their locations. In addition, contact constraints can also be handled as positional constraints once the body parts in contact were detected or specified. We compute the target locations of these parts, P_i, and define the positional constraints as:

$$C_P(\dot{q}_i) \quad = \quad (V'_{P,i} + J_{pos,i}\dot{q}_i) - P_i \tag{8}$$

where $J_{pos,i}$ is the Jacobian of the position derivative of each joint with respect to the joint angle derivatives, and $V'_{P,i}$ is the set of positions of the constrained joints at current frame.

C. Balancing Constraints

To keep the balance of the humanoid robot, a subset of physical constraints proposed in the Dynamics Filter [19] is enforced in the optimization. Specifically, the balance of the robot is maintained by controlling the full body posture such that the Center of Gravity (CoG) lies within the supporting surface on the ground.

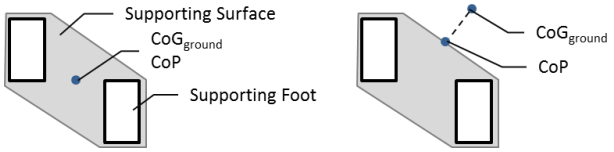

Fig. 3. The Center of Gravity projected on the ground and the Center of Pressure under (left) a balanced configuration, and (right) a potentially unbalanced configuration.

Here, we define the vertical projection of the CoG on the ground as CoG_{ground}. The supporting surface is evaluated as the area bounded by the planted foot/feet. The position of the closest point from the supporting surface to CoG_{ground} is assumed to be the Center of Pressure, CoP. Fig. 3 illustrates the relationship of CoG_{ground} and CoP. To maintain the stability of a motions, CoP is set as the target location of CoG_{ground}:

$$C_{CoP}(\dot{q}_i) = (CoG_{ground,i} + J_{CoGground,i}\dot{q}_i) - CoP_i \tag{9}$$

where $J_{CoGground,i}$ is the Jacobian of the position derivative of CoG_{ground} with respect to the joint angle derivatives, CoP_i is CoP at frame i.

D. Stepping Pattern Constraints

To avoid artifacts such as foot-sliding in the synthesized motion, as well as producing the stable gaits presented in the reference motion, we further introduce an additional constraint to enforce the stepping pattern in the reference motion. This is formulated as positional constraints for the joint vertices on the feet:

$$C_F(\dot{q}_i) = (V'_{feet,i} + J_{step,i}\dot{q}_i) - V^{ref}_{feet,i} \tag{10}$$

where $J_{step,i}$ is the Jacobian of the position derivative of each joint on the feet with respect to the joint angle derivatives, $V^{ref}_{feet,i}$ and $V'_{feet,i}$ are the position of each joint on the feet in the reference motion and the synthesized motion, respectively.

E. Constraint Type Decision

While it is possible to enforce all constraints in the optimization loop as the hard constraints explained in Section IV-D, the system may become over-constrained and result in unexpected behaviours. Our system allows the operator to switch some constraints into soft ones and use the energy term explained in Section IV-C to guide the optimization process.

In our system, we set the stepping pattern constraints hard, such that stable locomotion can be created and artifacts such as foot-sliding can be avoided. We also set collision constraints hard to avoid self-collisions and robot-environment collisions. The rest of the constraints are set as soft, such that they stabilize the motion while avoiding over-constrained. The weights W in Eq.4 are set 4.0 and 0.4 for the balancing and positional constraints respectively.

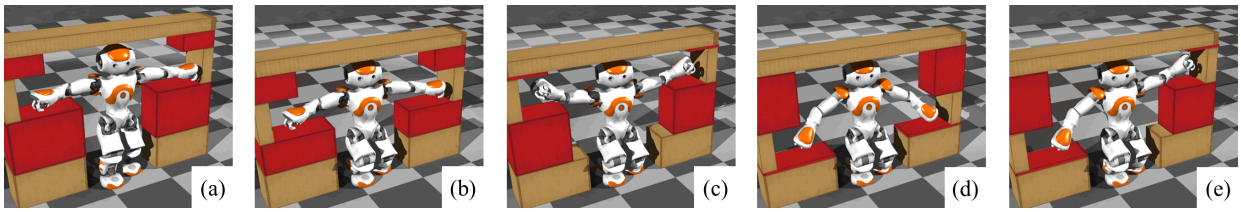

Fig. 4. The results to adapt the walking forward motion in a constrained environment.

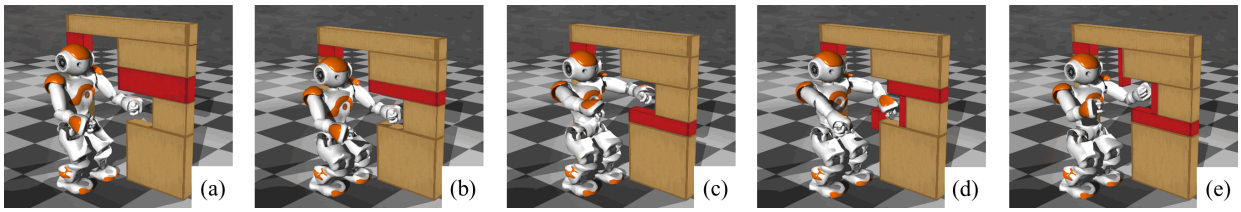

Fig. 5. The results to adapt the side stepping motion in a constrained environment.

VI. Experimental Results

In this section, we present the experiment results obtained from synthesizing humanoid robot movements by the proposed method. We simulate the movement of the Nao H21 V4.0 robot using the robot simulation software Webots [20]. The reference motions used in the experiments are edited based on the dynamically stable motions provided by Webots 6.4.4. The experiments were running on a computer with Intel Core-i7 Processor 3.40 GHz using single core. UMFPACK [21] is used as the linear solver.

A. Walking into Constrained Environment

In the first experiment, a walking forward motion of the Nao robot was designed as the reference motion. We set up a constrained environment for the Nao to pass through with its arms outstretched (Fig. 4 (a)). The spatial relationships between the Nao robot and the environment were extracted by the method explained in Section III. The Interaction Mesh consisted of 42 and 32 vertices sampled from the Nao robot and the objects in the environment respectively. To evaluate the effectiveness of the proposed method, we adjusted the size and shape of the objects in the environment so that the reference motion must be edited to avoid collisions.

Using our method, collision-free motions were produced automatically. In Test-A1 (Fig. 4 (b)), the height of the ceiling was lowered from $60cm$ to $51cm$, which was equivalent to 85% of the original height. The edited motion adapted to the changes of the environment by bending the knees to lower the body.

In Test-A2 (Fig. 4 (c)) and Test-A3 (Fig. 4 (d)), we modified the sizes of the boxes around the arms from Test-A1 such that the Nao robot had to raise and lower the arms by $15cm$ in order to pass through the aperture respectively. In Test-A4 (Fig. 4 (e)), we rearranged the boxes near to the arms based on the environment configurations in Test-A1 such that one arm needed to be raised and the other lowered to avoid collision.

For the stability of the synthesized motions, the CoG of the Nao robot is kept over the supporting surface in 98.46% of the synthesized postures. Because of the high percentage of stable postures, the Nao robot successfully kept the balance when walking through the constrained environments in the simulations. In terms of computational efficiency, the motion in each test contained 170 frames and required 32 seconds on average to synthesize the motion based on the altered environment. The computation cost is significantly smaller when comparing with other global path planning algorithms such as RRT, especially when the number of DOF is high.

B. Side Stepping into Constrained Environment

In the second experiment, a side stepping motion was designed for the Nao robot as the reference motion. The screen shot of the reference motion and the original environment configuration is shown in Fig. 5 (a). The Interaction Mesh consisted of 42 and 26 vertices sampled from the Nao robot and the objects in the environment respectively. Again, the size and shape of the objects in the environment were altered and the motion of the Nao robot was adapted accordingly.

In Test-B1 (Fig. 5 (b)), the height of the ceiling was lowered from $60cm$ to $52.9cm$, which was equivalent to 88% of the original height. The edited motion adapted to the changes in the environment by bending the knees and coordinating the body. In Test-B2 (Fig. 5 (c)), we modified the environment configurations from Test-B1 by raising the aperture around the arms by $10cm$. In Test-B3 (Fig. 5 (d)) and Test-B4 (Fig. 5 (e)), we further edited the shapes and sizes of the objects near to the arms based on the environment configurations in Test-B1, such that the arms needed to be bended downwards and upwards accordingly.

For the stability of the synthesized motions, the CoG of the Nao robot is kept over the supporting surface in 100% of the synthesized postures. For the computational cost, the motion in each test contained 720 frames and required 130 seconds on average to synthesize the motion based on the

altered environment. The readers are referred to the attachment video for the resulting motions.

Notice that while the height of the ceiling was reduced to 85% and 88% of the original height in the two experiments, the movement of the lower body had to be modified significantly. This was because the upper body had to maintain an upright position during the movements for better balancing. In particular, the height of the Nao robot's hip was reduced to 72.1% and 83.5% of its original value.

VII. Conclusions and Discussions

In this paper, we presented a spatial relationship based method to synthesize motions for controlling humanoid robots. Our method takes a reference motion and the configuration of a highly constrained environment as inputs, and produces collision-free and dynamically stable motion that adapts to the changes of the environment automatically. Experimental results show that our method can efficiently adapt the humanoid robot motions in significantly changed environments.

While we have successfully adapted the dynamically stable motion to different situations, there are some limitations. One of the assumptions in our method is that there is no external force applied to the robot except the ground reaction force. In case that there is any unexpected collision between the robot and the environment, the robot may lose balance and fall. One future direction is to incorporate a feedback system to deal with external perturbations and improve the robustness of the system.

Another limitation is that our method maintains the original stepping pattern. In case there are changes at the landscape, such as obstacles appearing on the floor, the strides have to be edited or even re-planned accordingly. This limitation also applies to adapting motions to non-flat terrains. Using the simplified balancing control in our method, it is difficult to produce physically stable motion on non-flat terrains without editing the stepping pattern of the robot.

For the future works, we are interested in designing a more dedicated foot step planner to manipulate the robot under complex terrains by changing the stepping pattern. In this research, we have only demonstrated the control of the Nao robot in our experiments. It would be interesting to extend our method for retargetting the reference motion to different humanoid robots with different joint configurations according to an altered environment. We did not experiment with real humanoid robots due to the limitation of research resources. Controlling real robots requires the consideration of run-time errors such as motor noises and unexpected collision. We would like to implement a feedback loop such that the system can re-plan the movement when unexpected situation occurs in the future. Finally, we would like to carry out in-depth comparisons on the performance of our proposed method with respect to global path planning algorithms.

Acknowledgement

The authors would like to thank the anonymous reviewers for their constructive comments and suggestions. They also thank Mr. Chris Blythe for his help in producing the demonstration video. This work was partially supported by the Hong Kong Baptist University RC start-up research grant.

References

[1] J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," apr 2000.

[2] J. Kuffner, "Dynamically-stable motion planning for humanoid robots," *Proceedings of IEEE International Conference on Humanoid Robotics*, 2000.

[3] S. Dalibard, A. Nakhaei, F. Lamiraux, and J.-P. Laumond, "Whole-body task planning for a humanoid robot: a way to integrate collision avoidance," in *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, dec. 2009, pp. 355 –360.

[4] D. Berenson, J. Chestnutt, S. Srinivasa, J. Kuffner, and S. Kagami, "Pose-constrained whole-body planning using task space region chains," in *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, dec. 2009, pp. 181 –187.

[5] K. Yamane, J. J. Kuffner, and J. K. Hodgins, "Synthesizing animations of human manipulation tasks," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 532–539, 2004.

[6] E. Ho, T. Komura, S. Ramamoorthy, and S. Vijayakumar, "Controlling humanoid robots in topology coordinates," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, oct. 2010, pp. 178 –182.

[7] E. S. Ho and T. Komura, "Character motion synthesis by topology coordinates," in *Computer Graphics Forum (Proceedings of Eurographics 2009)*, P. Dutr'e and M. Stamminger, Eds., vol. 28, no. 2, Munich, Germany, March 2009, pp. 299–308.

[8] NAO, "http://www.aldebaran-robotics.com/."

[9] D. Shinohara, T. Matsubara, and M. Kidode, "Learning motor skills with non-rigid materials by reinforcement learning," in *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*, dec. 2011, pp. 2676 –2681.

[10] T. Tamei, T. Matsubara, A. Rai, and T. Shibata, "Reinforcement learning of clothing assistance with a dual-arm robot," in *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, oct. 2011, pp. 733 –738.

[11] P. Vinayavekhin, S. Kudohf, and K. Ikeuchi, "Towards an automatic robot regrasping movement based on human demonstration using tangle topology," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, may 2011, pp. 3332 –3339.

[12] E. S. Ho, T. Komura, and C.-L. Tai, "Spatial relationship preserving character motion adaptation," *ACM Transactions on Graphics*, vol. 29, no. 4, pp. 1–8, 2010.

[13] D. Zarubin, V. Ivan, M. Toussaint, T. Komura, and S. Vijayakumar, "Hierarchical motion planning in topological representations," in *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

[14] H. Si and K. Gaertner, "Meshing piecewise linear complexes by constrained delaunay tetrahedralizations," in *Proc of the 14th International Meshing Roundtable*, 2005, pp. 147–163.

[15] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen, "Decimation of triangle meshes," *SIGGRAPH Comput. Graph.*, vol. 26, no. 2, pp. 65–70, Jul. 1992.

[16] M. Kim, K. Hyun, J. Kim, and J. Lee, "Synchronized multi-character motion editing," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 79:1–79:9, Jul. 2009.

[17] Y. Rubner, C. Tomasi, and L. J. Guibas, "A metric for distributions with applications to image databases," in *Proceedings of the Sixth International Conference on Computer Vision*, ser. ICCV '98. Washington, DC, USA: IEEE Computer Society, 1998, pp. 59–.

[18] R. Smith, "Open dynamics engine," 2007, http://www.ode.org/.

[19] K. Yamane and Y. Nakamura, "Dynamics filter - concept and implementation of online motion generator for human figures," *IEEE Transactions on Robotics*, vol. 19, no. 3, pp. 421–432, 2003.

[20] O. Michel, "Webots: Professional mobile robot simulation," *Journal of Advanced Robotics Systems*, vol. 1, no. 1, pp. 39–42, 2004.

[21] T. A. Davis, "Algorithm 832: UMFPACK, an unsymmetric-pattern multifrontal method," *ACM Transactions on Mathematical Software*, vol. 30, no. 2 (Jun.), pp. 196–199, 2004.