Hierarchical RRT for Humanoid Robot Footstep Planning with Multiple Constraints in Complex Environments

Hong Liu , Qing Sun and Tianwei Zhang

Abstract— **Humanoid robots have abilities of stepping over or onto obstacles, which is different from wheeled robots. However it may be difficult to apply the ordinary motion planning methods such as Rapidly-exploring Random Trees (RRT) to humanoid robots directly. Because these kinds of methods only consider to circumvent obstacles and ignore the constraint of balance. Aiming at dealing with these problems in one frame, a novel approach based on hierarchical RRT is used to plan the footstep for humanoid robots. It is designed according to three basic constraints: a transition model based gait generator, an inverted pendulum based balance controller and a collision detection based path planner. First, a set of layered transition model is utilized to revise the footstep according to the terrain condition, which is able to take full use of the motion ability as well as improve the efficiency. Then, a hierarchical strategy is exploited to select the feasible foot location to be added in the random tree based on the results of collision checking and balance control. Finally, a dynamic RRT method is introduced in our work to revise paths in changing environments. Different experiments are given to verify the feasibility and performance of the proposed approach in complicated environments with both dynamic and static obstacles.**

I. INTRODUCTION

Humanoid robotics is a hot research area during the past few decades. The popularity of humanoid robots is largely owing to their higher flexibility of action and better mobility. Compared with the wheeled robots, there are more choices for humanoid robots to deal with the obstacles in the path, because they are capable to step onto and over the obstacle in addition to dodging it. Thus ordinary methods of global navigation for mobile robots can not be implemented in the field of humanoid robots directly.

Motion planning for humanoid robots includes object grasping and manipulation, footstep placement and full-body motions, among which footstep planning is the basis to achieve other tasks [1]. A common way of computing global navigation strategies for biped humanoid robots is to generate a series of feasible foot placements[2]. It is implemented by using A* to search among a fixed set of transition model . However, the efficiency of the A* algorithm is limited by the size of the transition model. In a complex environment filled with different obstacles such as the scene in Fig 1, more predefined foot models are needed to adjust foot location for a specific action. A sampling-based method RRT is first introduced in the field of footstep planning in [3]. In their work, all of the successors are added to the tree in each extending process. With the referent footstep spreading over the whole search space, the method is effective in some specific environments, such as fields with local minima or narrow passages. However, the approach can not respond to the dynamic changes and does not take good use of the stepping capabilities. In this paper, a hierarchical RRT based method is used to search for the footstep displacement because of its efficiency in expanding the search space and capability to deal with large transition model. Main contribution of this paper can be concluded as follows:

1) The fixed set of transition model is replaced by a layered model in order to quickly adjust footstep in bad location.
2) A hierarchical strategy is added to the raw RRT in order to deal with large transition model and prevent the size of tree getting too large. Meanwhile, the modified RRT is executed with both constrains of balance control and collision detection.
3) A replanning is introduced to deal with changes in the environment by dynamically adjusting its initial paths.

The rest of this paper is organized as follows. Section II introduces the related work. Preliminaries are presented in Section III. In Section IV, a RRT-based footstep planning and some details of the algorithm will be presented. Experiments and discussion are introduced in Section V. Section VI draws the conclusions.

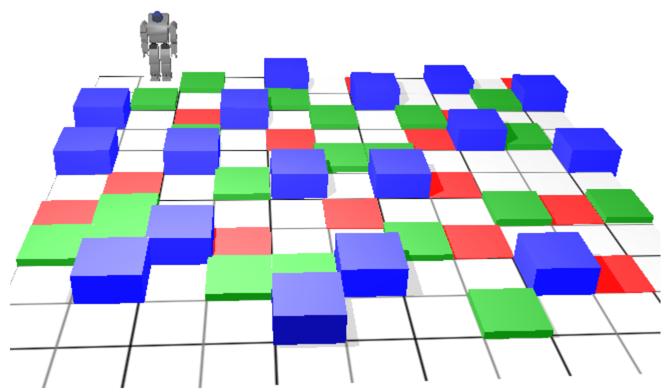


Fig. 1: An example of complicated environment

II. RELATED WORK

Due to the presence of sampling based methods such as PRM and RRT, the motion planning study has been

Hong Liu is with the Key Lab of Machine Perception and Intelligence, Shenzhen Graduate School, Peking University, China. hongliu@pku.edu.cn

Qing Sun is with the Key Lab of Machine Perception and Intelligence, Peking University, China. sunqing@cis.pku.edu.cn

Tianwei Zhang is with the Key Lab of Machine Perception and Intelligence, Shenzhen Graduate School, Peking University, China. zhangtianwei@sz.pku.edu.cn

improved significantly[4][5]. The traditional methods are usually realized by searching for a collision free path in the configuration space, which forces the robot to dodge the obstacles and our previous work are committed to improve the performance of these kinds of methods with some modifications[6][7][8]. However, for the humanoid robot, which is capable to step over or onto obstacles, these approaches fail to consider the additional abilities. Recently, techniques have been developed to generate the walking pattern for humanoid robot[9][10][11][12]. They are committed to address the problems of stable control and dynamic balance rather than searching for a collision-free path. In order to avoid collision with the obstacles, sensors are applied to perceive the environment information and navigate in unknown environment[13][14]. However, these methods do not consider the global information and may results in local minimal.

The concept of footstep planning for biped robots was first proposed in [4]. The algorithm uses a discrete set of predefined footstep locations that the robot can choose from for the next step. When each step pointing to an equal number of child nodes, a tree is generated from the initial footstep position. A* search is employed to find the best among the generated collision-free paths. The technique have been wildly applied to different robots in various kinds of environments[15][16][17]. There are improvements based on this approach including the application in some dynamic environments[18] and navigation for legged robots[19]. Ayaz *et al.* [20] presented a modified strategy which can distinguish the footstep taken to step over obstacles from the ordinary ones in obstacle cluttered environment. Garimort and Hornung apply D* Lite, a variation of A* for computing the optimal paths[21]. It enables the robot to reuse information when it has to revise its footstep plan according to changes in the environment. However, the efficiency of these approaches is limited by the size of transition models. As a result, they are not feasible in some specific environments.

The rapidly-exploring random tree (RRT) is first applied to humanoid footstep planning in [22]. The RRT have been shown to provide an efficient method for solving path planning problems for mobile robot and car with kino-dynamic constraints[23] [24][25]. Compared to the basic RRT method, which only extends the nearest node, all the actions in the transition model are added in the tree to explore the space in unfriendly space, such as fields with local minima or narrow passages.

In this paper, a modified RRT is proposed to hierarchically choose the feasible foot displacement from a layered set of transition model, which is effective in complex environments with both static and dynamic obstacles.

III. PRELIMINARIES

In this section, we will define the terms and notations used in this paper.

A. Basic RRT Algorithm

The rapidly-exploring random tree (RRT) is an incremental searching algorithm in the field of path planning. It initializes a searching tree with the initial configuration x_{init} as the root node. Then the tree spreads over the configuration space until it reaches the region within the threshold distance to the goal x_{goal}.The details of the basic RRT is outlined in Algorithm 1.

It is the common way to grow and explore the tree in the configuration space. However, it could result in very little expansion when applied to the area of footstep planning directly. In section 3, several extensions are introduced to improve the effectiveness of the search in cooperation with a layered transition model.

Algorithm 1 Basic RRT method

Input: tree T, x_{init} and x_{goal}

T.AddVertex(x_{init})
$x_{new} = x_{init}$
while Distance(x_{new}, x_{goal}) $>$ Threshold **do**
  x_{rand} = RandomConfiguration( )
  x_{near} = NearestNeighbor(x_{rand}, T)
  x_{new} = Extend(T, x_{rand}, x_{near})
  if x_{new} is collision free **then**
    T.AddVertex(x_{new}), T.AddVertex(x_{new}, x_{near})
  end if
end while
return T

B. Robot Model

The humanoid robot model used in our work is HOAP-2 by Fujitsu Automation. It is about 0.5 *m* high and weights 7 *kg*, with 25 degrees of freedom. The foot shape of HOAP-2 is a rectangle with 0.095 *m* in length and 0.065 *m* in width.

A Three-Dimensional Linear Inverted Pendulum Model (3D-LIPM) is used to simplify the complex full dynamic models. The rod corresponding to the swing leg is stretched with a stable and rotatable point on the ground and the mass point is linked to it. Let $c = (c_x, c_y, c_z)$ be the position of mass point and (p_x, p_y) represents the ZMP location on the floor, where x-y plane coincides with the horizontal plane. Take the property of 3D-LIPM and dynamical balance criterion into account, the relationship between the mass point and ZMP can be obtained as [9]:

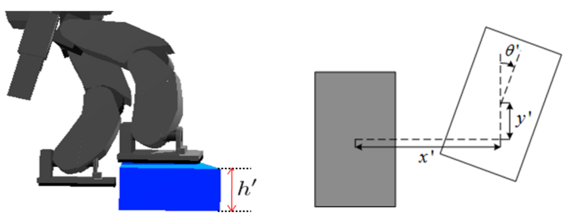
$$\ddot{c_x} = \frac{g}{z_c}(c_x - p_x) \tag{1}$$

$$\ddot{c_y} = \frac{g}{z_c}(c_y - p_y) \tag{2}$$

Here, z_c is the intercept of the plane where the simple model is constrained and g is the acceleration of gravity. For a given ZMP location, the related CoM trajectory can be obtained from the Eq.1 and Eq.2. Since the goal is to achieve a dynamically stable gait, the ZMP trajectory should always lie inside the supporting polygon, which is cited the balance criteria during planning. Once the movements of waist and feet are determined, the joints angle sequence can be obtained by inverse kinematics.

For the humanoid robot, planning the whole body navigation in the real world is computationally complex and not feasible, because many degrees of the robot have to be controlled. Thus a common way is to obtain a series of feasible foot locations in the real scenes with a set of predefined motions. Then the footsteps can be used to generate the stable walking pattern with the help of 3D-LIPM and ZMP. In the walking phase, two feet are alternatively served as the supporting foot to maintain the balance. The relationship between the supporting foot and swing foot can be defined by four parameters (x', y', θ', h'), such as shown in Fig 2, which represents the relative position x', y', orientation θ' and height h' between the two feet. Considering the joint limits violations and self-collisions, the reachable region is limited by the following parameters:

$$\psi_{left \rightarrow right} = \{x', y', \theta', h' \mid x' \in [-5cm, 12cm], \\ y' \in [6.5cm, 9.5cm], \theta' \in [-50°, 50°], h' \in [-5cm, 5cm]\} \tag{3}$$



(a) (b)

Fig. 2: The footstep model

C. Layered Transition Model

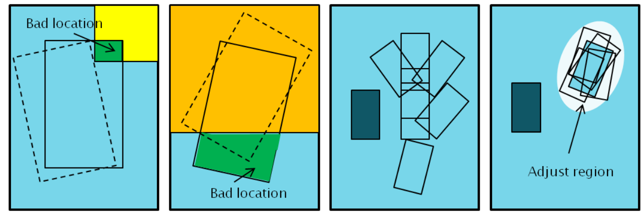
A finite set of predefined actions, denoted as τ, is always used to plan the footstep in a continuous state space, which represent the next possible foot location. Thus, the foot model can also be expressed as:

$$\tau = (x', y', \theta') \tag{4}$$
$$\psi = (\tau, h') \tag{5}$$

It is worth mention that the actions of transition model determine the reachable region for every step. When the scene is simple with few obstacles, a set of basic foot actions are enough to deal with the planning during walking. However, when the scene is filled with various kinds of obstacles, more particular actions are needed in order to step onto or circumvent obstacles. For these reasons, the related actions τ are classified into two classes: the basic walking motion set Γ_b and adjustive motion set Γ_a. For each basic action, there is a subset of adjustive motions and they both construct the predefined footstep transition model:

$$\Gamma_b = (\tau_i | i = 1, 2, ..., n) \tag{6}$$
$$\Gamma_i = (\tau_{i(k)} | i = 1, 2, ..., m) \tag{7}$$
$$\Gamma_a = \Gamma_1 \cup \Gamma_2 ... \cup \Gamma_n \tag{8}$$
$$\Gamma_{all} = \Gamma_b \cup \Gamma_a \tag{9}$$

Here, τ_i presents the basic action and $\tau_{i(k)}$ denotes the adjustive motion related to each basic one, the number of which is n and m respectively. Example of transition model can be seen in Fig.3(c) and Fig.3(d).



(a) (b) (c) (d)

Fig. 3: (a) and (b) are all bad locations where the foot collides with obstacle or lands on a unstable position respectively. (c) basic action set (d) adjust action $\tau_{i(k)}$ within the adjust region

The adjustive action is designed to expand the stepping capabilities in bad terrain, such as the situation in Fig 3(a) and Fig 3(b). Thus, the robot does not need to take additional steps when stepping on the stairs or moving in tight area. In order to avoid collisions with obstacle or bad location from different directions, the adjust actions are designed around each basic footstep action within a certain region:

$$\Omega_i = \begin{cases} |x_i'(k) - x_i'| & < \alpha \\ |y_i'(k) - y_i'| & < \beta \\ |\theta_i'(k) - \theta_i'| & < \gamma \end{cases} \tag{10}$$

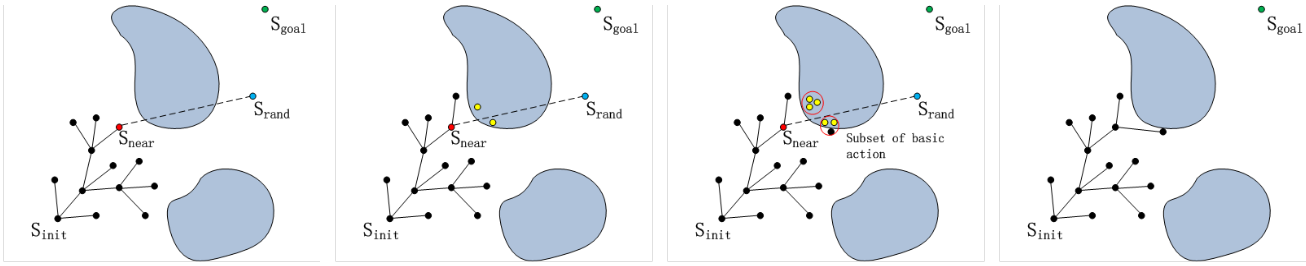
Here, Ω_i denotes the adjustive region for a particular motion τ_i. α, γ and β indicate the threshold of position and rotation between τ_i and $\tau_{i(k)}$.

D. Environment Model

An equally-sized grid map is used to represent the environment information in our work. Each cell contains the related global position, a height value and a flag marked as either free or occupied:

$$E = \{x, y, h, tag | (x, y, h) \in \mathbb{R}^3, tag \in \{0, 1\}\} \tag{11}$$

From this map, we are able to compute the support region by eliminating the relevant cells underneath the foot. The collision checking used in our work is a polygon-polygon intersection test between the outline of foot geometry and the outline of obstacle projection onto the walking surface. The outline of obstacle is a little enlarged to avoid the robot getting too close to the obstacles. Based on the dynamic balance criterion for humanoid robots, which means the ZMP should lie inside the support region during the walking process, we can test whether a location is stable or not by calculating the position between ZMP trajectory and the support polygon. To make the walking look more natural, a moving ZMP reference is used to generate the walking pattern [26]. In the single support phase, ZMP is set along the middle axis of supporting foot from back to front, the boundary of which has the same distance to the center; While, in the double support phase, ZMP shifts from one foot to the other smoothly.



(a) (b) (c) (d)

Fig. 4: The process of RRT based footstep planning. (a) Given a partial RRT being grown from an initial foot state to the goal state, a node is randomly selected with goal-biased. The nearest node in the tree is selected to extend out towards the sample node. (b) Basic actions are selected for next foot extension based on the heuristic function rules. (c) Adjust action set are added to the tree when the basic one is not feasible. (d) With the feasible foot transition being added, the tree is expanded in each iteration.

Algorithm 2 Hierarchical RRT for footstep planning

Input: Tree T, Γ_{all} , s_{init} and s_{goal}

1: T.AddVertex(s_{init})
2: $s_{new} = s_{init}$
3: **while** Distance(s_{new}, s_{goal}) $>$ Threshold **do**
4: $p =$ Random(0,1)
5: **if** $p >$ goalProbility **then**
6: $s_{rand} = s_{goal}$
7: **else**
8: $s_{rand} =$ RandomConfiguration( )
9: **end if**
10: $s_{near} =$ NearestNeighbor(s_{rand}, T)
11: $s_{new(i)} =$ GetPossibleFoot(s_{near},Γ_b)
12: **for** $i = 1$ to k **do**
13: **if** ($s_{new(i)}$ is valid) **then**
14: T.AddVertex($s_{new(i)}$)
15: **else**
16: $P =$ AlignExtend($s_{new(j)}$, Γ_i)
17: for each element $s_{new(j)}$ in P
18: **if** $s_{new(j)}$ is valid **then**
19: T.AddVertex($s_{new(j)}$)
20: BREAK
21: **end if**
22: **end if**
23: **end for**
24: if no node is valid
25: delete s_{near} from the tree
26: **end while**
27: **return** T

IV. RRT BASED FOOTSTEP PLANNING

We use RRT as the searching method for footstep planning because of its efficiency in exploring the configuration space and the ability to deal with a large size of transition model. In this section details are given on how the hierarchical RRT algorithm is introduced to the footstep planning with multiple constraints.

A. The Algorithm

The search space for growing the RRT tree is a content of foot locations in the world coordinate system. Each state s is parameterized by the global position(x, y, h), orientation θ and supporting foot mark δ :

$$s = \{x, y, h, \theta, \delta | \delta \in \{L, R\}\} \tag{12}$$

For each supporting foot state s, the next state is determined by the function f of motion related to it:

$$s' = f(s, \psi) \tag{13}$$

Given an environment map and a set of layered transition model, A hierarchical RRT is executed to compute a series of feasible foot locations. The details of the algorithm are presented in Algorithm 2. The tree starts from the node of initial foot. Rather than sampling the node completely randomly, a goal-based control is used to make the growth of tree more efficiently towards the goal region: with probability $1 - p$, the node is randomly sampled by the function RandomConfiguration( ): with probability p, the node q_{rand} is set to the goal. Once the sample node has been generated, the nearest neighbor of q_{rand} in the tree is extended out towards it(line10). The most important modification is how to select the node for extension, which is shown from line 11 to 20. The details of the node extending process will be described later. And the whole process of the algorithm is presented in Fig 4

B. Hierarchical Strategy to Extend Node

The node extension process determines how the tree can be expanded. On account of the discrete set of footstep state, the common way to expand the tree with only one successor may not be feasible in difficult area. In [22], all of successors are added to the tree with a goal-biased modification. However, when the size of transition model gets large, the complexity to compute the nearest neighbor for each node and the time of collision checking quickly increase.

Moreover, it is useless to extend all the actions since only a small part of them are effective at each step. We should trade off between the size of transition model and the efficiency of the method. In order to make full use of the motion abilities The extension process should locate the foot into an reachable region as well as control the size of expanded tree in each planning loop. Thus, a hierarchical extending strategy is proposed in our work. At first, a set of basic actions are obtained by the function GetPossibleFoot( ) and added to the tree if they are valid. This is the first layer of extension (line 11-14). The second layer is not needed unless the basic foot collides with obstacles or has a bad location. Then, the second layer of extension would be called by the function AlignExtend( ), which selects node from the adjustive actions for the related basic one. Here a blind search is used, which is implemented by randomly choosing $\tau_i(k)$ from Γ_i. We do not compute which action is exactly the best for the current condition as it is computationally complex and not feasible. If no nodes are added in the tree in the two layered extension, the nearest node will be deleted from the tree and a new circle of iteration starts until the tree grows to the region of goal. The state will be added in the tree if it is valid in the configuration space. The judgement of whether a state is valid includes two parts: whether the foot location is collision free with obstacles and whether it is stable with a reasonable support region.

The function GenerateExtend() returns a set of basic action nodes rather than one node, because it is difficult to decide which is best for the current state with the consideration of both local condition and global information. Three possible ways to expand the tree in the first layered extension have been identified:

(1)Quickly expanding orientation: the state which is the nearest to the random configurations

$$h1 = w_1||(x, y) - (x_{rand}, y_{rand})|| + w_2||\theta - \theta_{rand}|| \quad (14)$$

This is the common for the basic RRT to extend nodes, which is considered as a Monte-Carlo way of biased searching into the largest Voronoi regions.

(2)Goal orientation: the state which is nearest to the goal configurations

$$h2 = w_1||(x, y) - (x_{goal}, y_{goal})|| + w_2||\theta - \theta_{goal}|| \quad (15)$$

Adding the state which is closer to goal enables the planner to consider global information into and explore with efficiency.

(3)Gait smooth orientation: the state with minimal changes

$$h3 = w_3||h - h_{near}|| + w_2||\theta - \theta_{near}|| \quad (16)$$

Here, we want to minimize the difference between the extended state and the previous one by selecting the node with minimal changes.

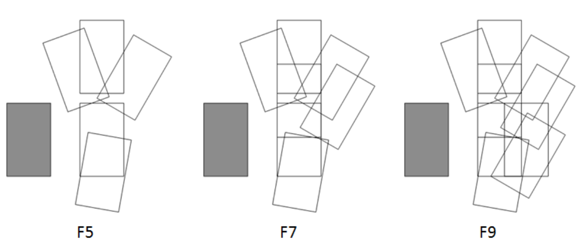


Fig. 5: Basic footstep sets with 5, 7, 9 different footsteps, shown here relative to the left supporting foot (black).

C. Replanning

The above algorithm does not consider the dynamic change in the environment, which may be common in our dairy life. For example, the robot may slippage onto the ground or encounter some unpredictable interruption. Then replanning is needed to adapt the footstep path to the new situations. An advantage of RRT method is that there are some variations to adaptive for different area, one of which is DRRT [24]. DRRT is a probabilistic analog to the widely-used D* family of deterministic replanning algorithms. It is implemented by efficiently removing just the newly-invalid parts and maintain the rest. At first, a path is generated off-line according to the initial position of all the obstacles. Once changes are detected to interrupt the previous path, the replanning is rapidly called to plan the new paths from the current position to goal position. Since D-RRT plans in the reverse order from the goal, we have to change for the start and goal node in each replanning iteration. At the same time, the grid map is updated together with the dynamic obstacles.

V. EXPERIMENTS AND ANALYSIS

To evaluate the proposed method, lots of simulation experiments are implemented by Webots, which is a simulator that allows users to simulate dynamic behaviors of robots in a 3D virtual environment. The model used for humanoid robot is HOAP-2. The simulation program is running under Windows 7 on an AMD 8600 processor with 2 G Byte of memory. In order to test the performance of the proposed method, experiments are performed within different scenarios.

The tested environment are a static indoor environment, an dynamic outdoor environment and a scenario filled with different kinds of obstacles. Because the node is randomly sampled in the RRTs method, all the RRT methods are tested 100 times for each environment and the results present the average of all the tests. And the results include the average number of the expanded state(for A* and D* Lite), sampling nodes, nodes in the tree, collision detection calls , total time and steps needed to generate the paths.

Sets of basic motions are very important in our work, which influence the planning behavior and results. First, the basic motions has to guarantee the regular walking states,

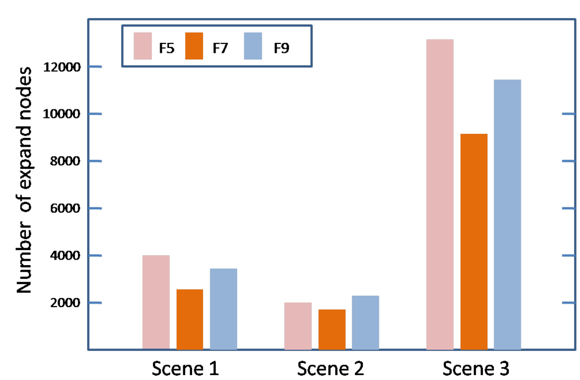


Fig. 6: Average number of expanded nodes for basic footstep sets in different environments.

TABLE I: PARAMETERS OF THE LAYERED TRANSITION MODEL

Γ_b	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7
τ_1 $(x'(m), y'(m), \theta)$	(0.08, 0.1, 0)	(0.08 0.05 0)	(0.08, 0, 0)	(0.07, 0.07, -30)	(0.09, 0.07, 30)	(0.09, 0, 30)	(0.08, -0.4, 0)
ω_i $(\alpha(cm), \beta(cm), \gamma)$	(0.8, 1.5, 25)	(1, 2, 30)	(0.8, 2, 20)	(0.5, 1, 10)	(0.5, 1, 20)	(1, 2.5, 20)	(0.5, 1, 15)
k (number of adjust)	6	6	4	6	8	6	4

such as going straight, turning left, turning right, standing still, stepping back and etc. Then, for each basic motion, the adjustive action should satisfy the Eq.(3) and cover the reach region as much as possible. We test three sets of basic footsteps(Fig. 5) in different kinds of environments and the comparison result can be seen in Fig. 6. According to the average number of expanded nodes, footstep set F7 yields efficient plans in our experiment and the parameters are shown in Table I. It is worth to mention that further analysis and improvements of the parameters can probably help to obtain faster results based on our algorithm, which is an interesting issue to be studied in the future. The weighting factors used for the cost function were $w1 = 1.0, w2 = 0.2$, and $w3 = 0.1$. All of these values were determined experimentally, and offered reasonable results.

A. Static Indoor Environment

Fig. 7: Static indoor environment

In the first scenario(Fig. 7), the robot has to move from sofa in one room to the front of desk in another room. The challenge of the scenario includes a narrow passage to walk through and the stairs with several steps. The obstacles are all static, such as tea table, chairs, sofa, glove box and etc. The environment is tested with 4 different methods. In order to test the effect of the layered transition model, the model used for A* algorithm and Multiple RRT is fixed, similar to related references[15] and [3]respectively. The results of different methods in this environment are show in Table II, which indicate that Hierarchical RRT with goal-biased control is able to solve the query much faster and with relatively fewer steps. For A* method the query time increases as many steps are needed to find its way through tight areas. The multiple-RRT has the same problem with limited transition models and more steps are needed to put the robots correctly on the stairs. We also test multiple-RRT with the layered transition

TABLE II: RESULTS FOR STATIC INDOOR ENVIRONMENT

Method	$Time$	N_{sample}	N_{expand}	$Steps$
A* method	2.68	–	6784	50
Basic RRT	7.53	8796	8796	53
Multiple RRT	2.35	536	5358	49
HRRT	2.04	1968	4356	47
HRRT ($p = 0.1$)	1.78	1258	2548	45

model, which needs more than 10s to find out the path. In the penultimate row, Hierarchical RRT is executed without goal biased control. In the last row, the RRT is executed with a goal probability, which is set to 0.1. Although the results in Table II indicates that goal-based HRRT seems to be effective in finding the path, it not always performs better. Fig. 8 gives results of different values of goal probability test in this environment, from which we can find that the planning time quickly when probability exceeds the optimal value. It is worth to mention that when the value is larger than 0.8, the RRT planner can not find a path.

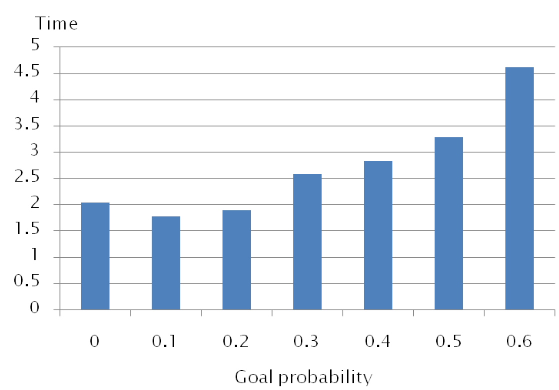


Fig. 8: Average planning time for different values of goal-probability threshold

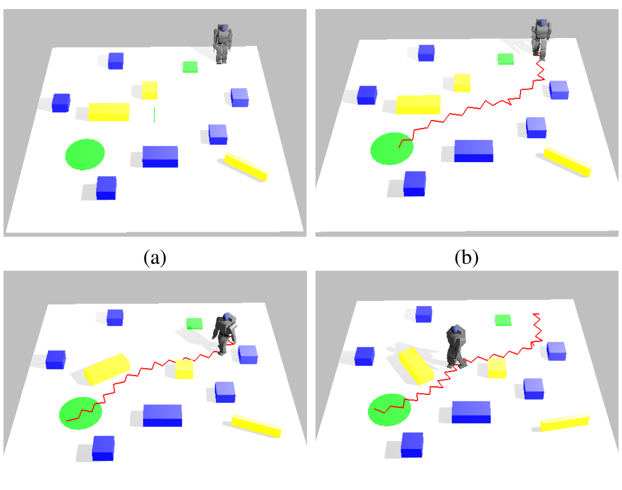
B. Dynamic Environment

In the second scenario(Fig. 9), three kinds of obstacles are placed randomly in the environment with the area of $1.5m \times 1.5m$. The blue ones presents the static obstacles the robot has to circumvent, the green one are those lower obstacles for robot to step on and the yellow ones are the dynamic obstacles. The dynamic obstacles are either moving forward or rotating about the axis. Here, we assume that the 2D grid of environment is given and updated when the position of obstacles changes. Two different methods are compared here, both of which are able to deal with the dynamic environment. The average results of 100 experiments in this environment

TABLE III: RESULTS FOR DYNAMICAL ENVIRONMENT

Method	$T_{initial}$	N_{expand}	$S_{initial}$
D* Lite	1.28	3874	29
Hierarchical RRT	1.09	1347	30
Method	T_{replan}	N_{expand}	S_{replan}
D* Lite	0.78	2621	21
Hierarchical RRT	0.57	674	18
Method	T'_{replan}	N'_{expand}	S'_{replan}
D* Lite	0.31	1620	15
Hierarchical RRT	0.28	469	14

are listed in Table III. The efficiency of planning in dynamic environment is determined by the planning time. Since the change is unpredictable, the robot has to quickly compute the replanning path. Otherwise, the replanned path is useless to adapt for the new situation. From the table , it is clear that our methods is able to explore this environment more effectively than the D* Lite footstep planning. This is mainly caused the lower obstacles, which may be a challenge.



(a) (b) (c) (d)

Fig. 9: Planning result by our method in dynamic environment (a)The environment (b)Initial planning path (c) Replanning for the first time (d) Replanning for the second time

C. Discrete Cluttered Environment

In the third scene, we make the scene much complicated to test the effectiveness of our algorithm in the cluttered environment. The ground are separated by different kinds of cubes with the same size. The black, green and white cubes are save for the robots to step onto with the height of $0.1m$, $0.05m$ and $0m$ respectively. However, the robot has to circumvent the black ones because the height is beyond the limit of it. The Challenge for the robot is the multiple times of stepping up and down in order to reach the goal. With a adjust model of 40 actions, our method is able to plan the stable footstep sequence in average 12.8s. As a comparison, other methods including A* and multiple RRT based footstep planning need more than 30s to generate the whole paths. Fig 10 gives the results of our method in this environment and some details of the foot location can been seen from Fig 11.

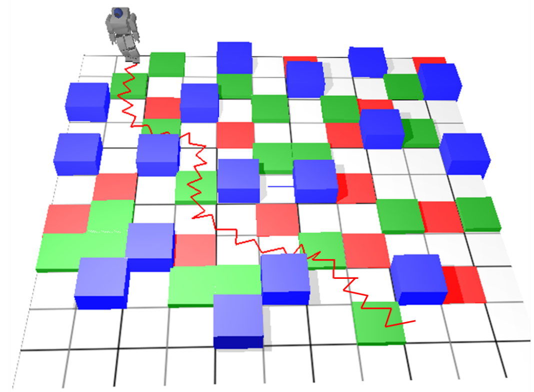


Fig. 10: Example of path generated by our method in discrete cluttered environment

VI. CONCLUSIONS

In this paper, a method based on Hierarchical RRT is proposed to solve the problem of footstep planning for humanoid robots in complex environments. This approach expands the reachable region and provides more motion for next step with a layered transition model. And the hierarchical strategy results in faster planning times, by simplifying sequences of action needed to fit through narrow passages in the search space. Furthermore, additional abilities are considered with the constraints of balance control and collision detection. In different kinds of environments, the proposed method provides very promising results, which can not only be effective in dealing with different obstacles and narrow passages, but also adaptive to changes in environments.

In the future, we plan to implement the proposed methods on real physical robot. Much work needed to been done for improvement of the frameworksuch as optimization of the layered transition model and some extensions on uneven terrain.

VII. ACKNOWLEDGMENTS

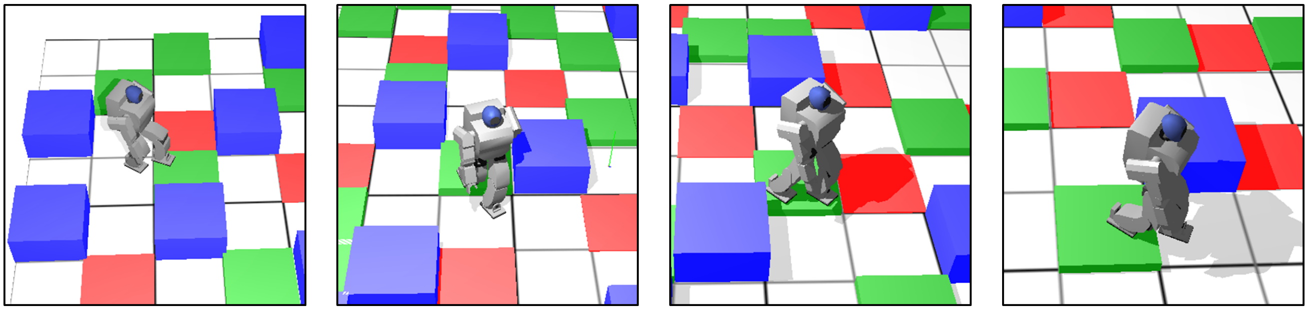
This work is supported by National Natural Science Foundation of China(NSFC, No.60875050, 60675025), National High Technology Research and Development Program of China(863 Program, No.2006AA04Z247), Scientific and Technical Innovation Commission of Shenzhen Municipality (No.JC201005280682A, CXC201104210010A).

REFERENCES

[1] J. Kuffner, K. Nishiwaki, S. Kagami2, M. Inaba, and H. Inoue ”Motion Planning for Humanoid Robots”, in *International Symposium on Robotics Research*, pp. 629, 2003.

[2] J. Kuffner, K. Nishiwaki and S. Kagami, “Footstep Planning Among Obstacle for Biped Robots”, in *IEEE International Conference on Intelligent Robots and Systems*, pp. 500-505, 2001.

[3] Z. Xia, J. Xiong and K. Chen, “Global Navigation for Humanoid Robots Using Samling-Based Footstep Planners”, *IEEE Transactions on mechatronics*, Vol 16, No 4, pp. 716-723, August 2011



(a) (b) (c) (d)

Fig. 11: Details of the foot location in discrete cluttered environment

[4] S. M. LaValle and J. J. Kuffner, “Rapidly-exploring random trees: Progress and prospects, *in Workshop on Algorithmic Foundation of Robotics*, 2000.

[5] L. E. Kavraki, P. Svestka, J. C. Latombe and M. H. Overmars, “Probabilistic roadmaps for fast path planning in high-dimensional configuration spaces, *in IEEE Transactions on Robotics and Automation*, pp. 566-580, 1996.

[6] H. Liu, W. W. Wan and H. B. Zha, ” A dynamic subgoal path planner for unpredictable environments”, in *IEEE International Conference on Robotics and Automation*, pp. 994-1001, 2010.

[7] H. Liu, H. Wen and Y. Li, ”Path planning in changing environments by using optimal path segment search”, in *IEEE International Conference on Intelligent Robots and Systems*, pp. 1439-1445, 2009.

[8] H. Liu, D. Ding, W. W. Wan and Hongbin Zha, ”Predictive model for path planning by using k-near dynamic bridge builder and Inner Parzen Window”, in *IEEE International Conference on Intelligent Robots and Systems*, pp. 2133-2138, 2008.

[9] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi and H. Hirukawa, “Biped Walking Pattern Generation by Using Preview Control of Zero-Moment Point”, in *IEEE International Conference on Robotics and Automation*, pp. 1620-1626, 2003.

[10] K. K. Noh, J. G. Kim and U. Y. Huh, “Stability Experiment of a Biped Walking Robot with Inverted Pendulum”, in *the IEEE Conference of Industrial Electronics Society*, pp. 2475-2479 , 2004.

[11] K.Erbatur, A. Okazaki, K. Obiya, T.Takahashi and A. Kawamura, “A Study on the Zero Moment Point Measurement for Biped Walking Robots”, in *International Workshop on Advanced Motion Control*, pp. 431-436, 2002.

[12] C.Zhu, Y. Tomizawa, X. Luo and A. Kawamura, “Biped Walking with Variable ZMP, Frictional Constraint, and Inverted Pendulum Model”, in *IEEE International Conference on Robotics and Biomimetics*, pp. 425-430, 2004.

[13] M. Yagi and V. J. Lumelsky, ”Biped robot locomotion in scenes with unknow environment”, in *IEEE International Conference on Robotics and Automation*, pp. 375-380, 1999.

[14] R. Cupec, G. Schmidt, O. Lorch, ”Experiments in Vision-Guided Robot Walking in a Structured Scenario”, in *IEEE International Symposium on Industrial Electronics*, pp. 1581-1586, 2005.

[15] J. Chestnutt, M. Lau, G. Cheung, J. Kuffner, J. Hodgins and T. Kanade, “Footstep Planning for the Honda ASIMO Humanoid”, in *IEEE International Conference on Robotics and Automation*, pp. 629, 2005.

[16] R. Ozawa, Y. Takaoka, Y .Kida, K. Nishiwaki, J. Chestnutt, J. Kuffner, S. Kagami, H. Mizoguch and H. Inoue, “Using Visual Odometry to Create 3D Maps for Online Footstep Planning”, in *IEEE International Conference on Systems, Man and Cybernetics*, PP. 2643-2648, 2005.

[17] J. Chestnutt, Y. Takaoka, M. Doi, K. Suga and S. Kagami, “Safe adjustment regions for legged locomotion paths”, in *IEEE International Conference on Humanoid Robots*, pp. 224-229, 2010.

[18] M. Philipp, J. Chestnutt, J. Kuffner and T. Kanade, “Vision-Guided Humanoid Footstep Planning for Dynamic Environments”, in *IEEE International Conference on Humanoid Robots*, pp. 13-18, 2005.

[19] J. Chestnutt, “Navigation Planning for Legged Robots” [PhD. Thesis], *CMU-RI-TR-56-23, Nov.2007*, The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania USA.

[20] Y. Ayaz, K. Munawar, M. B. Malik, A. Konno, M. Uchiyama, “Human-Like Approach to Footstep Planning Among Obstacles for Humanoid Robots”, in *IEEE International Conference on Intelligent Robots and Systems*, pp. 5490-5495, 2006.

[21] J. Garimort, A. Hornung, M. Bennewitz, ”Humanoid navigation with dynamic footstep plans”, in *IEEE International Conference on Robotics and Automation*, pp. 3982 - 3987, 2011.

[22] Z. Xia, G. Chen, J. Xiong, Q. Zhao, K, Chen, ”A random sampling-based approach to goal directed footstep planning for humanoid robots”, in IEEE International Conference on Advanced and Intelligent Mechatronics, pp. 168 - 173 , 2009.

[23] M. Kobilarov and G. Sukhatme, ”Time optimal path planning on outdoor terrain for mobile robots under dynamic constraints”, University of Southern California Technical Report CRES-04-009, 2004.

[24] D. Ferguson, N. Kalra and A. Stentz, ”Replanning with RRTs”, in *IEEE International Conference on Robotics and Automation*, pp. 1243-1248, 2006.

[25] M. Likhachev and D. Ferguson, ”Planning long dynamically-feasible maneuvers for autonomous vehicles” in *International Journal of Robotics Research*, vol. 28, no. 8, pp. 933-945, 2009.

[26] C.Zhu, Y. Tomizawa, X. Luo and A. Kawamura, “Biped Walking with Variable ZMP, Frictional Constraint, and Inverted Pendulum Model”, in *IEEE International Conference on Robotics and Biomimetics*, pp. 425-430, 2004.