A Workspace Decomposition for Hierarchical Motion Planning with Humanoid Robots

Yong-Tae Kim[†‡], Salvatore Candido[‡] and Seth Hutchinson[‡]

[†]Dept. of Information and Control Engineering
Hankyong National University
Anseong, Gyeonggi, 456-749 Korea
ytkim@hknu.ac.kr

[‡]Dept. of Electrical and Computer Engineering
University of Illinois
Urbana, IL 61801 USA
{candido,seth}@uiuc.edu

***Abstract*— This paper presents a hierarchical motion planner for humanoid robots navigating complex environments. We use a workspace decomposition that allows our planning algorithm to be separated into high-level, subgoal, and local algorithms. The workspace decomposition consists of a passage map, obstacle map, gradient map, and local map. We verify our approach using a virtual humanoid robot in a simulated environment.**

I. INTRODUCTION

Wheeled mobile robots navigate in a two dimensional world and usually have only three degrees of freedom. The configuration space of wheeled mobile robots can be searched efficiently by probabilistic, sample-based motion planning methods. Bipedal robots such as humanoids can walk in a three dimensional workspace. A typical biped robot has twelve degrees of freedom in its lower body and more in the upper body. The high degree of actuation gives the robot a wide range of motions. Humanoid robots can walk, turn, step on/down stairs [1], [2], [3], [4], [5], [6], [7], step over barriers [8], climb over small obstacles [4], [5], [7], [9], [10], climb ladder [10], and crawl through narrow passages [11], [12].

Motion planning for humanoid robots in a 3D workspace is both a computationally expensive and theoretically challenging problem. One difficult aspect of this problem is that humanoid robots have a configuration space that is generally higher than six dimensions. Due to the many actuators of humanoid robots, it is difficult to find feasible motions in complex environments with various kinds of obstacles. Also, searching in higher dimensional configuration spaces requires significant memory and computation time. Because of the complexity of the configuration space and the robot's stability requirements, it becomes difficult to find collision free trajectories in cluttered environments.

One strategy that simplifies the motion planning problem for humanoid robots is to use preplanned motions that begin and end with a statically stable pose [1], [13], [10]. Much research on generation of stable and efficient gaits for humanoid robots has been conducted considering dynamic stability constraints [6], [14]. These patterns of motion can be reused in sequence to convert one or more preplanned steps into a walking motion. In this paper, we refer to these preplanned motions as *motion primitives*. If these preplanned motion primitives are optimized, the resulting motions can be more efficient and robust; using minimal computational expense at run time. By describing the motion of the robot as a set of motion primitives and using a set of heuristics to validate footstep locations, the planner can find a path through difficult environments [13].

Motion planning for a humanoid robot on flat ground can be considered as a search for a sequence of feasible motion primitives rather than a search of a high dimensional configuration space for a trajectory [1], [13]. However, in indoor environments with gates, obstacles, stairs, inclines, holes and barriers, motions of humanoids are constrained and the motion planner should be designed with many motion primitives to overcome various situations. Also, searching the entire high dimensional configuration space is needed. Therefore, the planner may not be able to find a feasible sequence of primitives in a short amount of time.

A hierarchical, multi-level planning approach can be applied to solve this problem [4], [16], [17]. The high-level planner only considers global path planning; finding a rough sketch of the global path by treating the humanoid robot as an appropriate bounding volume. In high level planning, it is important to generate a global navigation map that is built considering the motion primitives available to the robot. The mid-level planning chooses appropriate subgoals along the global path specified by the high level planner. The subgoals on the global path can reduce the search space for a local motion planner and are useful for stable navigation by the robot. Choosing desirable subgoals is the challenging part of the hierarchical planning. The low-level planner provides a sequence of motion primitives to reach the subgoal specified by the mid-level planner.

This paper proposes a hierarchical planning strategy based on a combination of 2.5 dimensional maps, a decomposition of the 3D workspace. Using these maps, hierarchical motion plans are made that allow the humanoid robot to navigate complex environments using only a small set of motion primitives. At first, we define *locomotion primitives* based on the motion primitives of the robot. The high-level global planner finds a path by using a global navigation map that is obtained from a passage height map, an obstacle height map and a gradient height map of obstacles. In the mid-level planning, the subgoals where the robot stably copes with various obstacles are decided based on the gradient map of obstacles. Subgoals are useful for stable navigation of the robot and allow the local planner to use only a small set of locomotion primitives. The low-level planner determines an optimal sequence of locomotion primitives in proximity to subgoals by using the A* algorithm. To reduce the search space, we use a local search guided by a global path and a matching algorithm of locomotion primitives. We verify our approach on a virtual humanoid robot with a simulated environment. Simulation results show a short planning time for feasible paths in cluttered environment.

II. PROBLEM FORMULATION

A. Configuration of Humanoid and Locomotion Primitives

We define a pose, $\mathbf{q}$, to be a set of the joint angles of the m actuators in the humanoid robot $\mathcal{H}$. A configuration γ of the robot $\mathcal{H}$ is a point in its configuration space $\mathcal{Q}$ where $\mathcal{Q} = SE(3) \times \mathbb{T}^m$ parameterized by

$$\gamma = (x_r, y_r, z_r, \phi_r, \theta_r, \psi_r, \mathbf{q}). \tag{1}$$

The robot's global location and orientation are specified by comparing a world reference frame to a coordinate frame fixed rigidly to itself. This frame is referred to as the robot's root.

Because the humanoid robot has a $6 + m$ dimensional configuration as in Eq. 1, it is difficult to find a path through 3D cluttered environments and check for collisions with the obstacles. Also, searching in the high dimensional configuration space needs a significant amount of memory and computation time and may not be able to find a solution. To simplify a motion planning problem of the robot, we define motion primitives and locomotion primitives for the humanoid robot.

Let a *motion primitive* of the robot $\mathcal{H}$ be a pattern of motion that begins and ends with a statically stable pose. It is stored either as an ordered set of poses with a defined interpolation or as an ordered set of inputs to the robot's controller :

$$P_i = (\mathbf{q}_{i1}, \mathbf{q}_{i2}, \mathbf{q}_{i3}, \cdots \mathbf{q}_{in_i}) \tag{2}$$

where n_i is the number of poses for *i-th* primitive, $\mathbf{q}_{i1}$ is the initial pose and $\mathbf{q}_{in_i}$ is the final pose of the robot executing the primitive. The motion primitives that are efficient and robust, and look natural can be precomputed [10].

A *locomotion primitive* of the robot $\mathcal{H}$ is denoted by a parameterization of a motion primitive as

$$\mathbf{P}_i = (\Delta x_r, \Delta y_r, \Delta z_r, \Delta\theta_r, \Delta\phi_r, \Delta\psi_r, P_i, \mathcal{C}_{pi}, V_{p_i}). \tag{3}$$

Δx_r, Δy_r, and Δz_r are the net displacements of the origin of the robot's root frame in local coordinates, $\Delta\theta_r$, $\Delta\phi_r$, and $\Delta\psi_r$ are the net changes in orientation of that frame, P_i is a motion primitive, $\mathcal{C}_{pi}$ is the overall cost of execution time, distance of displacement, consumed energy and potential instability for the locomotion, and V_{p_i} is a volume containing the swept volume of the locomotion. The swept volume V_{p_i} is used for fast collision checking in the low-level planner. Thus, a motion primitive P_i specifies a sequence of internal configuration changes while a locomotion primitive $\mathbf{P}_i$ describes the motion of the robot in its ambient space.

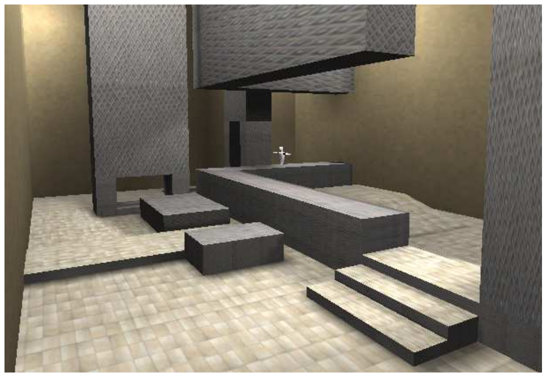


Fig. 1. Virtual humanoid robot and 3D Workspace.

The set of all locomotion primitives is,

$$\mathbf{P} = \{\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3, \cdots, \mathbf{P}_L\}, \tag{4}$$

where L is a number of locomotion primitives.

The configuration of the robot is changed through the application of the locomotion primitives that are chosen by the planner. A primitive, $\mathbf{P}_i \in \mathbf{P}$, or a sequence of primitives, $\pi = (\mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_K)$, when applied to the current configuration, γ, produces a new configuration, γ'. A *configuration transition function*, f is specified by

$$f : (\gamma,\ \pi) \ \mapsto\ \gamma'. \tag{5}$$

The task of a motion planner is to find a sequence of locomotion primitives that when applied, transform the start configuration to some goal configuration.

B. Workspace Model and 2.5 Dimensional Maps

A humanoid robot operates in a 3D workspace, $\mathcal{W} \subseteq \mathbb{R}^3$. We assume the workspace can be partitioned into a floor, gates, obstacles, stairs, inclines, holes and barriers. The floor is assumed to be a horizontal and even plane. Low-height obstacles and holes will have steep side walls and are sparse. Basically, these assumptions correspond to smooth, indoor environments. In this paper, we consider motion planning of humanoid robots in the complex indoor environment as shown in Fig. 1.

To represent the 3D workspace, we use a combination of 2.5 dimensional maps of its environment. The obstacle height map M_{O_h} stores the height of the floor or of an object sitting on the floor with respect to the floor plane.

$$M_{O_h} : (x, y) \mapsto o_h \ , \ o_h \in R \tag{6}$$

where x, y is 2D plane of workspace and o_h is a height of the obstacle. From the obstacle height map, we can generate the gradient height map of obstacle as

$$M_{\nabla O_h} : (x, y) \mapsto (\frac{\partial o_h(x, y)}{\partial x}, \frac{\partial o_h(x, y)}{\partial y}). \tag{7}$$

We use the gradient height map to obtain the information of edge and border of the obstacle.

To climb an incline plane and maintain stability, the robot should apply a locomotion primitive with the same orientation as the incline. The obstacle orientation map can be obtained from the workspace as

$$M_{O_o} : (x, y) \mapsto (\phi_o, \psi_o) \ , \quad \phi_o, \psi_o \in [-\pi, \pi] \tag{8}$$

where ϕ_o and ψ_o are the tilting angles, pitch and roll, of the obstacle. We can get the gradient orientation map of obstacle as

$$M_{\nabla O_o} : (x, y) \mapsto (\frac{\partial \phi_o}{\partial x}, \frac{\partial \phi_o}{\partial y}, \frac{\partial \psi_o}{\partial x}, \frac{\partial \psi_o}{\partial y}). \tag{9}$$

The robot cannot walk through the passage with a low ceiling and a low gate. The ceiling height with respect to the floor plane is stored in the ceiling height map:

$$M_{C_h} : (x, y) \mapsto c_h \ , \quad c_h \in R^+ \tag{10}$$

where c_h is a height of ceiling. From the obstacle height map $M_{O_h}$and the ceiling height map M_{C_h}, we can obtained the passage height map as

$$M_{P_h} : (x, y) \mapsto p_h \ , \quad p_h \in R^+ \tag{11}$$

where $p_h(x, y) = c_h(x, y) - o_h(x, y)$.

We model the 3D environment by a combination of six 2.5 dimensional maps that are used to generate a global navigation map for the global path planning and a local obstacle map for the sub-goal planning.

$$\mathcal{W}_{2.5D} = \{M_{O_h}, M_{\nabla O_h}, M_{O_o}, M_{\nabla O_o}, M_{C_h}, M_{P_h}\}.$$

III. HIERARCHICAL MOTION PLANNING

In complex environments as shown in Fig. 1, the motion planner should have various kinds of locomotion primitives, e.g., walking, turning, stepping on/down and stepping over, to cope with the various obstacles, maintain the stability of the robot, and find a feasible sequence of primitives. We design a hierarchical motion planner to solve the planning problem of humanoid robot in the complex 3D workspace.

A. Global Path Planner

The task of the high-level planner is only to find a global path between an initial configuration and a goal configuration in the complex environment. We assume that the humanoid robot $\mathcal{H}$ is a mobile robot with a bounding volume, which is obtained from the set of locomotion primitives $\mathbf{P}$. In the high-level planning, it is important to create a global navigation map that describes only *hard obstacles*, which forbid the movement of the robot. The robot $\mathcal{H}$ can cope with *easy obstacles* such as stairs, low obstacles, gates, holes, inclines and low barriers. The planner generates a 2D global navigation map and finds a global path based on the global navigation map.

At first, we consider the vertical obstructions that prevent the robot from moving. The vertical hard obstacles are decided based on a minimum height, k_p, of the pathway that the robot can move through. The 2D global passage map is obtained from the passage height map M_{h_p} :

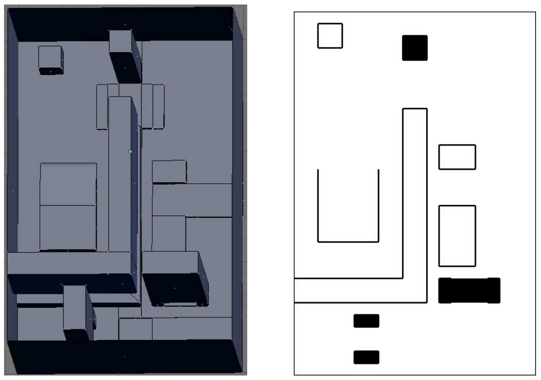

(a) Top view of workspace. (b) Global navigation map.

Fig. 2. Global navigation map.

$$M_{gp}(x, y) = \begin{cases} 1, & if \ \ M_{h_p}(x, y) \ < \ k_p, \\ 0, & otherwise. \end{cases} \tag{12}$$

Fig. 2(a) is a top view of the indoor environment shown in Fig. 1. Also, the hard obstacles are decided by using both a steepness and a height difference of the object, which can be obtained from the gradient height map of obstacle and the obstacle height map. Let $N(x, y)$ be a neighborhood around a position, $(x, y) \in \mathbb{R}^2$, and $\Delta M_{O_h}(x, y)$ be a maximum height difference in $N(x, y)$:

$$\Delta M_{O_h}(x, y) = \max_{(x', y') \in N(x, y)} |M_{O_h}(x, y) - M_{O_h}(x', y')|. \tag{13}$$

Then we can generate a 2D global obstacle map using the equation:

$$M_{go}(x, y) = \begin{cases} 1, \ if \ \|M_{\nabla O_h}(x, y)\| > k_s \\ \qquad and \ \ |\Delta M_{O_h}(x, y)| > k_h, \\ 0, \ otherwise, \end{cases} \tag{14}$$

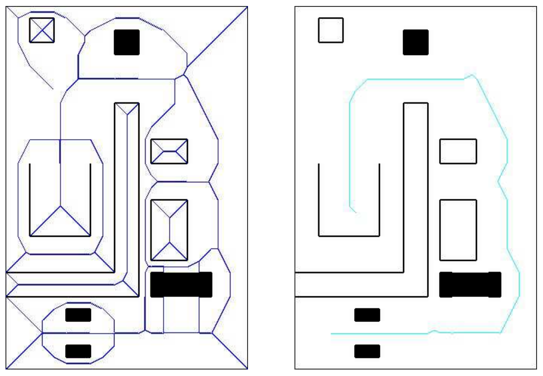
where k_s is a steepness value that can classify the hard obstacles and k_h is a maximum height difference that robot can step on or over. The global obstacle map shows the edges or borders of the hard obstacles.

From the global passage map and the global obstacle map, we can obtain a 2D global navigation map for the global path planner, which is shown in Fig. 2(b).

$$M_{gn}(x, y) = \begin{cases} 1, \ if \ M_{gp}(x, y) = 1 \ \textbf{or} \ M_{go}(x, y) = 1 \ , \\ 0, \ otherwise. \end{cases} \tag{15}$$

$\mathcal{W}_{free} = \{(x, y) | M_{gn}(x, y)(x, y) = 0\}$ is a free workspace that the humanoid robot can move.

Given an initial configuration $\gamma_{initial}$, a goal configuration γ_{goal} and a 2D navigation map M_{gn}, the task of the global path planner is to find a 2D global optimal path τ_g that is selected amongst a set of feasible paths. Because the navigation map M_{gn} is two dimensional, we consider the x and y position of the configuration and simplify the robot as a circle enclosing the projected area of the minimum bounding volume for the robot $\mathcal{H}$. Since the optimal motion



(a) Skeleton using NF2. (b) A resulting global path.

Fig. 3. A global path planning using Wavefront/NF2 algorithm.

plans that find shortest Euclidean path bring the robot too close to obstacles and allow the robot to touch the obstacle vertices, we use a maximum clearance planning method, Wavefront/NF2 algorithm [18], to find a global path that has a maximum clearance with obstacles. Fig. 3 shows the resulting skeleton and a global path using Wavefront/NF2 planning algorithm.

B. Sub-Goal Planner

The objective of the mid-level sub-goal planner is to find the appropriate subgoals along the global path specified by the global path planner. Selecting the optimal subgoals is a important part of the hierarchical planner. The humanoid robot can not step on the edges of objects and may need many locomotion primitives to overcome the easy obstacles. Choosing correct subgoals near the easy obstacles can help the robot to cope with the easy obstacles and keep stable navigation in complex environments. Also, it can decrease the number of overall locomotion primitives for the robot and reduce the search space for the local motion planning.

To obtain the information of overall objects in the environment, we create a local obstacle map. Edges and borders of the obstacles can be extracted from the gradient height and orientation maps of obstacles. From those maps, we generate a 2D local obstacle map as

$$M_{lo}(x,y) = \begin{cases} 1, & if\ \|M_{\nabla O_h}\| > k_s\ and\ \|M_{\nabla O_o}\| > k_o, \\ 0, & otherwise, \end{cases} \tag{16}$$

where k_o is a minimum tilting value that classifies the inclines. $\mathcal{W}_{edge} = \{(x,y)|M_{lo}(x,y) = 1\}$ is the area that is composed of edges or borders of the obstacles. The humanoid robot can not step on, but can step over the area, $\mathcal{W}_{edge}$. Fig. 4 shows the resulting local obstacle map with the edge information of all the objects in the environment.

The basic idea of sub-goal planning is to select the sub-goals near the easy obstacles that the robot should pass through. Because each sub-goal configuration becomes an initial configuration of the next planning step in the local motion planning, the robot can overcome the obstacle with only a locomotion primitive. Therefore, the sub-goal planning can reduce the number of locomotion primitives and

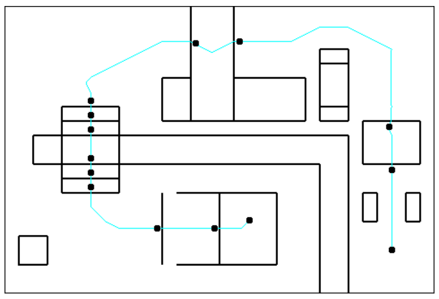


Fig. 4. Local obstacle map and resulting sub-goals.

searching space for the low-level motion planning.

To find the sub-goals near the easy obstacles, we search the first junction points, **a**, where the global path meets with an edge in the area, W_{eb}, from a initial point to a goal.

$$\mathbf{a} = \{a_1, a_2, \cdots, a_J\} \subseteq \{(x,y)|\mathcal{W}_{edge} \cap \tau_g\}, \tag{17}$$

where J is the number of junction points. Because these junction points can not be selected as sub-goals, we search the sub-goal candidates near the junction points. One candidate of each junction point a_i is the front point c_{f_i} on the global path, which satisfies the equation:

$$\|c_{f_i} - a_i\| = k_f,\ for\ i = 1, 2, \cdots J, \tag{18}$$

where k_f is a minimum distance that humanoid robot needs to apply the locomotion primitives near easy obstacles. We can get a set of front candidates of the sub-goals, $\mathbf{c}_f = \{c_{f_1}, c_{f_2}, \cdots, c_{f_j}\}$. Also, we can find a set of rear candidates of the sub-goals, $\mathbf{c}_b = \{c_{b_1}, c_{b_2}, \cdots, c_{b_j}\}$, that are the rear points of each a_j on the global path and satisfy the equation, $\|c_{b_i} - a_i\| = k_b,\ for\ i = 1, 2, \cdots J$.

In most cases, we can select a front candidate c_{f_i} as a sub-goal. However, when the width of an obstacle is small and the robot $\mathcal{H}$ can step over the obstacle, we should select a rear candidate c_{b_i} as a sub-goal. That is, if the distance between sub-goals is less than a maximum distance k_r that the robot can stably walk through, step over or on, i.e., $\|c_{f_i} - c_{b_{i+1}}\| < k_r$, we select the rear candidate $c_{b_{i+1}}$ as sub-goal instead of the front candidate $c_{f_{i+1}}$. By using this algorithm, we can obtain a set of sub-goals, $\mathbf{c} = \{c_1, c_2, \cdots, c_J\}$, where $c_i \in \mathbb{R}^2$.

Fig. 4 shows the resulting sub-goals that are planned based on the local obstacle map and the global path specified by the global path planner. The sub-goal planner sends the resulting sub-goals, c_i and c_{i+1}, and a local path τ_l to the low-level motion planner. The local path τ_l is a path that starts from c_i to c_{i+1} along the global path τ_g,

$$PathSegment(\tau_g, c_i, c_{i+1}) \mapsto \tau_l.$$

C. Local Motion Planner

The task of low-level planner is the local motion planning that searches for a sequence of locomotion primitives between subgoals along a local path. An initial configuration of the robot is obtained based on a sub-goal c_i as

$$\gamma_{init} = \{c_i, M_{O_h}(c_i), \angle c_{f_i} c_{b_i}, M_{O_o}(c_i), \mathbf{q}_s\}, \tag{19}$$

where $\angle c_{f_i} c_{b_i}$ means an angle of the line connecting between a front candidate c_{f_i} and a rear candidate c_{b_i} of the sub-goal position c_i. $M_{O_h}(c_i)$ and $M_{O_o}(c_i)$ are a height and tilting angles of the obstacle at the sub-goal. $\mathbf{q}_s$ is an upright standing pose, which is the same initial pose as all primitives that are designed to cope with all the easy obstacles. Similarly, a goal configuration of the robot, γ_{goal}, is generated based on the other sub-goal c_{i+1}.

Then, the purpose of the local motion planner is to provide an optimal sequence of locomotion primitives π^* from the initial configuration γ_{init} to the goal configuration γ_{goal} along the local path τ_l using the workspace model $\mathcal{W}_{2.5D}$ of the environment:

$$LocalPlanner(\gamma_{init},\ \gamma_{goal},\ \tau_l,\ \mathcal{W}_{2.5D})\ \mapsto\ \pi^*, \tag{20}$$

where $\pi^* = (\mathbf{P}_1^*, \mathbf{P}_2^*, \cdots, \mathbf{P}_K^*)$ and $\gamma_{goal} = f(\gamma_{init}, \pi^*)$. The initial configuration γ_{init} has the same initial pose as the first primitive $\mathbf{P}_1$ and the final pose of a primitive $\mathbf{P}_i$ should be identical with the initial pose of the next primitive $\mathbf{P}_{i+1}$.

Let π_i be a feasible sequence of locomotion primitives, $(\mathbf{P}_{i1}, \mathbf{P}_{i2}, \cdots, \mathbf{P}_{iK_i})$, that satisfies $\gamma_{goal} \approx f(\gamma_{init}, \pi_i),\ i = 1, 2, \cdots, K_N$. The local motion planner chooses an optimal sequence π^*, amongst the K_N feasible sequences, that minimize the cost function $\mathcal{C}$:

$$\pi^* = \arg\min_{\pi_i}\ \ \mathcal{C}(\gamma_{init},\ \tau_l,\ \pi_i). \tag{21}$$

The following cost function is considered.

$$\mathcal{C}(\gamma_{init}, \tau_l, \pi_i) = \sum_{j=1}^{K_i}\ [\mathcal{C}_{P_{ij}} +\ \rho(f(\gamma_{init}, \pi_{ij}), \tau_l)], \tag{22}$$

where $\pi_{ij} = (\mathbf{P}_{i1}, \mathbf{P}_{i2}, \cdots, \mathbf{P}_{ij})$. $\mathcal{C}_{P_{ij}}$ is a cost of the locomotion primitive P_{ij} and the cost ρ is a minimum Euclidean distance between a configuration and the local path.

We employ the A* algorithm to find an optimal sequence π^* as shown in Fig. 5. The nodes of the queue are sorted based on both a cost-to-come value, $\mathcal{C}_{P_i} + \rho$, and a cost-to-go value, h, the Euclidean distance between the current configuration and the goal configuration of the robot.

An important part of the planning algorithm is to obtain the admissible locomotion primitives along the local path. The admissible primitives of the robot are selected by the algorithm as shown in Fig. 6.

```
LocalPlanner($\gamma_{init},\ \gamma_{goal},\ \tau_l,\ \mathcal{W}_{2.5D}$)
$Q$ – the queue of explored nodes
$n$ – $\{\gamma_n, \mathcal{C}_n, \pi_n\}$
 1 $n_{start} \leftarrow \{\gamma_{init}, \emptyset, \emptyset\}$
 2 $Q \leftarrow \{n_{start}\}$
 3 repeat
 4     if $Q = \emptyset$ return failure
 5     sort $Q$ by $\mathcal{C}_n + h(\gamma_n, \gamma_{goal})$
 6     $n \leftarrow Q.minimum()$
 7     if ($\|\gamma_n - \gamma_{goal}\| < \epsilon$)
 8         return $\pi_n$
 9     else
10         $\mathbf{P}_{admissible} \leftarrow$ FindLocomtion($\gamma_n, \tau_l, \mathcal{W}_{2.5D}$)
11         for all $\mathbf{P}_i \in \mathbf{P}_{admissible}$
12             $n_c \leftarrow \{f(\gamma_n, \mathbf{P}_i),\ \mathcal{C}_n + \mathcal{C}_{P_i} + \rho,\ \pi_n \otimes \mathbf{P}_i\}$
13             $Q \leftarrow Q \cup \{n_c\}$
```

Fig. 5. Local motion planning using the A* algorithm.

```
FindLocomtion($\gamma_n,\ \tau_l,\ \mathcal{W}_{2.5D}$)
1 $\mathbf{P}_{admissible} \leftarrow \emptyset$
2 for all $\mathbf{P}_i \in \mathbf{P}$
3     if (Transition($\gamma_n, \mathbf{P}_i$) $= 1$) and ($\mathbf{P}_i$ not in collision)
4         $\gamma_n' \leftarrow f(\gamma_n, \mathbf{P}_i)$
5         if (Near Neighbor($\gamma_n', \tau_l$) $= 1$)
6             if (Height Slope Match($\gamma_n', \mathcal{W}_{2.5D}$) $= 1$)
7                 $\mathbf{P}_{admissible} \leftarrow \mathbf{P}_{admissible} \cup \{\mathbf{P}_i\}$
8 return $\mathbf{P}_{admissible}$
```

Fig. 6. Searching algorithm of admissible locomotion primitives.

IV. SIMULATION RESULTS

To verify our approach, we use a simulated humanoid robot with twenty- eight degrees of freedom. The simulations were performed utilizing Crystal Space, a 3D SDK [19], on a 3 GHz Pentium 4 with 1 GB of memory. Our maps were constructed in 1-2 seconds. Building the decomposition, forming a high level plan, and finding subgoals took less than 10 seconds. The local plans (A* searches) took on average 6.2 seconds in the example in Fig. 7 and Fig. 8. Note that the time to execute a local search is largely dependent on the local environment and set of primitives. Also, many optimizations for the individual components of the planners are available.

Fig. 1 shows a snapshot of the robot's workspace. The high level planner creates a path through the global navigation map, which constrains the movement of the robot only by obstacles that it cannot step on or down into. Fig. 2 shows the navigation maps generated for our simulated workspace. Fig. 3 shows the (a) skeleton function and (b) global path generated by the NF2 algorithm.

The subgoal planner selects the required points the robot must reach at the end of the execution of a primitive in order to make the global path feasible. Fig. 4 shows the global path superimposed on the local obstacle map and the subgoals identified in our environment.

The local planner has access to twenty-one primitives including walking, walking with a turning motions, shuffle steps, step up and down, and primitives to walk on a ramp. All primitives are assumed statically stable and the transitions are at statically stable poses. Fig. 7 and Fig. 8 show some results from our simulation environment. Traversing the path shown required around 150 steps.

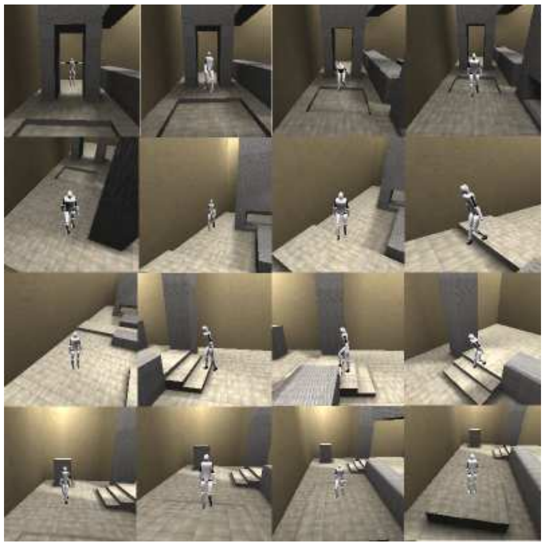


Fig. 7. Snapshots of simulation result.

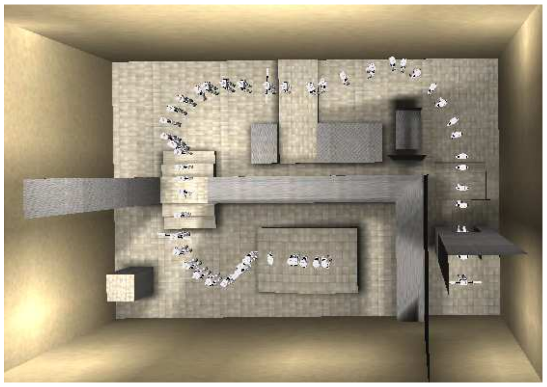


Fig. 8. Top view of simulation result.

V. CONCLUSIONS AND FUTURE WORK

We present a hierarchical motion planning method that, using an intelligent workspace decomposition, finds a global path and transforms it to a sequence of motion primitives. We design a planner based on motion primitives and use a hierarchical structure to reduce the search space of the planner. Using subgoals found on the global path, the proposed planner decreases the number of required motion primitives and efficiently copes with an environment that is cluttered with various kinds of the obstacles. We verify our approach on a virtual humanoid robot with simulated environment. Simulation results show a fast planning time and an stable motion quality.

It seems that the proposed method can be extended to the workspace that has uneven terrain and a layered environment. We will design both a global navigation map and a local obstacle map, and develop a real-time motion planner for humanoid robots moving in these types of workspaces.

REFERENCES

[1] J.J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue, ”Dynamically-stable motion planning for humanoid robots”, *Autonomous Robots*, Vol. 12, No. 1, pp. 105-118, 2002.

[2] J.-S. Gutmann, M. Fukuchi, and M. Fujita, “Stair climbing for humanoid robots using stereo vision”, *In Int. Conf. on Intelligent Robots and Systems (IROS)*, Sendai, Japan, 2004.

[3] J.-S. Gutmann, M. Fukuchi, and M. Fujita, “ Real-time path planning for humanoid robot navigation”, *In Int. Joint Conference on Artificial Intelligence (IJCAI)*, Edinburgh, Scotland, 2005.

[4] J. Chestnutt and J.J. Kuffner, “A tiered planning strategy for biped navigation.”, *In Int. Conf. on Humanoid Robotics (Humanoids)*, 2004.

[5] J. Chestnutt, J.J. Kuffner, K. Nishiwaki, and S. Kagami, “Planning biped navigation strategies in complex environments”, *In Int. Conf. on Humanoid Robotics (Humanoids)*, 2003.

[6] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka, “The development of Honda humanoid robot”, *In Int. Conf. on Robotics and Automation (ICRA)*, pp. 1321-1326, May 1998.

[7] O. Lorch, A. Albert, J. Denk, M. Gerecke, R. Cupec, J. F. Seara, W. Gerth, and G. Schmidt, “Experiments in vision-guided biped walking”, *In Int. Conf. Intelligent Robots and Systems (IROS)*, 2002.

[8] K. Okada, S. Kagami, M. Inaba, and H. Inoue, “Plane segment finder: Algorithm, implementation, and applications”, *In Int. Conf. on Robotics and Automation (ICRA)*, Seoul, Korea, May 2001.

[9] Y. Guan, K. Yokoi, N.E. Sian, and K. Tanie, “Feasibility of humanoid robots stepping over obstacles”, *In Int. Conf. on Intelligent Robots and Systems (IROS)*, Sendai, Japan, 2004.

[10] K. Hauser, T. Bretl, J.-C. Latombe, “Using Motion Primitives in Probabilistic Sample-Based Planning for Humanoid Robots”, *In proceedings of the Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2006.

[11] F. Kanehiro, T. Yoshimi, S. Kajita, M. Morisawa K. Fujiwara, K. Harada, K. Kaneko, H. Hirukawa, and F. Tomita, “Whole body locomotion planning of humanoid robots based on a 3D grid map”, *In Int. Conf. on Robotics and Automation (ICRA)*, Barcelona, Spain, 2005.

[12] Z. Shiller, K. Yamane, and Y. Nakamura, “Planning motion patterns of human figures using a multi-layered grid and the dynamics filter”, *In Int. Conf. on Robotics and Automation (ICRA)*, Seoul, Korea, 2001.

[13] J. J. Kuffner, Jr., K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue. “Motion planning for humanoid robots”, *In Int. Symp. Rob. Res.*, Siena, Italy, 2003.

[14] K. Nagasaka, M. Inaba, and H. Inoue, ”Walking pattern generation for a humanoid robot based on optimal gradient method”, *In Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics*, 1999.

[15] Y. Kroki, M. Fujita, T. Ishida, K. Nagasaka, and J. Yamaguchi, “A small biped entertainment robot exploring attractive applications”, *In Int. Conf. on Robotics and Automation (ICRA)*, 2003.

[16] T.-Y. Li, P.-F. Chen, and P.-Z. Huang, “Motion planning for humanoid walking in a layered environment”, *In Int. Conf. Robotics and Automation (ICRA)*, 2003.

[17] J.-S. Gutmann, M. Fukuchi, and M. Fujita, “A Modular Architecture for Humanoid Robot Navigation”, *In Int. Conf. on Humanoid Robotics (Humanoids)*, pp. 26-31, 2005.

[18] J.-C. Latombe, *Robot Motion Planning*, Kluwer, Boston, MA, 1991.

[19] http://www.crystalspace3d.org/